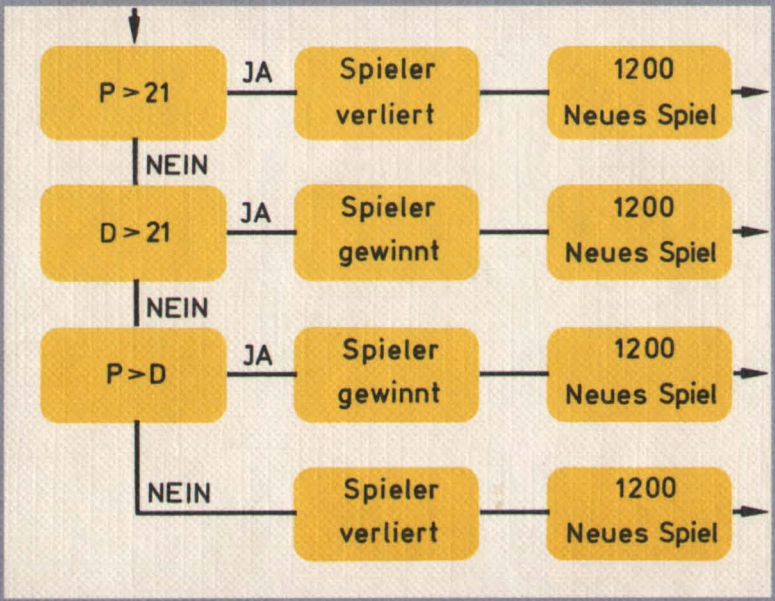
Franzis
Computer-Praxis

Busch
Basic für Einsteiger

Der leichte Weg zum selbständigen Programmieren



Bausteine eines Computers
Daten – Bits – Bytes
Wie rechnet ein Rechner
Der Computer als Rechenmaschine
Nun stellt der Computer die Fragen
Der Computer als Entertainer
Der Computer liest Daten
Der Computer als Digitaluhr
Sie planen mit dem Computer die Investitionen
Der Computer als Textautomat
Lagertechnik
Die Logik der Steuerungstechnik
Der Computer als Vermögensberater
Der Computer als Sortiermaschine
Der Computer zum Spielen
Spezielle Tastenfunktionen

Franzis'

Basic für Einsteiger

Rudolf Busch

Basic für Einsteiger

Der leichte Weg zum selbständigen Programmieren

Mit 32 Abbildungen

© 1984 Franzis-Verlag GmbH, München
Lizenzausgabe mit Genehmigung des Franzis-Verlages GmbH, München
für die Bertelsmann Club GmbH, Gütersloh
die Europäische Bildungsgemeinschaft Verlags-GmbH, Kornwestheim
die Buchgemeinschaft Donauland Kremayr & Scheriau, Wien
und die Buch- und Schallplattenfreunde GmbH, Zug/Schweiz
Diese Lizenz gilt auch für die Deutsche Buch-Gemeinschaft
C. A. Koch's Verlag Nachf., Berlin – Darmstadt – Wien
Einbandgestaltung: Erich Gebhardt
Einbandfoto: Alfred Pasieka, Hilden
Druck und Bindung: Mohndruck Graphische Betriebe GmbH, Gütersloh
Printed in Germany · Titel-Nr.: 00553 8

Vorwort

Was ist ein Mikroprozessor? Wie arbeitet ein Mikrocomputer? Auf diese Fragen findet man heute viele Antworten. Arbeitsplatzvernichter, „Jobkiller"... sagen die einen. Andere meinen, daß ohne diese modernen Bauelemente und Geräte der Elektronik die Aufgaben der Zukunft nicht bewältigt werden können. Die obigen Antworten spiegeln Meinungen, Gefühle; auch Ängste wieder – und diese kommen vom „nicht-genau-Wissen".

Sie haben sich für „wissen-wollen" entschieden – und Sie tun gut daran. BASIC wollen Sie lernen, eine der zahlreichen Programmiersprachen, die Sie in die Lage versetzt, aus dem stummen Kasten mit dem Namen „Mikrocomputer" erst ein nützliches Hilfsmittel zur Lösung Ihrer Probleme zu machen.

Im Verlaufe dieses Buches gehe ich davon aus, daß Sie bei der Lektüre Ihren Computer vor sich haben, um alle beschriebenen Programmschritte gleich auszuprobieren. Sicherlich ist das die effektivste Methode, in die neue Materie einzudringen.

Aber auch wenn Sie (noch) nicht einen „Personal-Computer" Ihr eigen nennen: ich habe mich bemüht, Text und Beispiele so abzufassen, daß Sie auch ohne Übungen lernen, was BASIC ist und wie man damit umgeht. Vielleicht ist dann das Buch eine Entscheidungshilfe und Sie erfüllen sich endlich einen langgehegten Wunsch...

Oder – auch das will ich nicht ausschließen – Sie erkennen, daß das Programmieren doch schwieriger ist, als Sie vermutet hatten. Dann habe ich Sie wenigstens vor einer teuren Fehlinvestition bewahrt.

Wie auch immer – ich habe nicht den Ehrgeiz, Ihnen hier BASIC in allen möglichen Variationen und mit den letzten Raffinessen vorzuführen. Ohne Schnörkel und Verzierungen, unter Verzicht auf das imponierende – oder ängstigende – Fachchinesisch werden Sie lernen, BASIC zu handhaben.

Es wird Ihnen gefallen, mit relativ wenigen Befehlen virtuos zu arbeiten; besser, als einen nur blassen Schimmer des Gesamten zu bekommen. Sie werden sich noch wundern, was Sie Ihrem Computer alles entlocken können.

Sie brauchen nicht zu wissen, wie der Computer „das macht". Aber schaden tut's auch nicht. Ich konnte mir deshalb nicht verkneifen, Ihnen in den ersten Kapiteln etwas über das Innenleben des Computers zu erzählen. Wenn Sie das

schon wissen, – oder wenn es Sie nicht interessiert – dann beginnen Sie die Lektüren einfach erst auf Seite 20.

Und nun genug der Vorrede – fangen wir an!
Sie werden Ihren Spaß haben!

Gifhorn, 1984 Rudolf Busch

Wichtiger Hinweis

Die in diesem Buch wiedergegebenen Schaltungen und Verfahren werden ohne Rücksicht auf die Patentlage mitgeteilt. Sie sind ausschließlich für Amateur- und Lehrzwecke bestimmt und dürfen nicht gewerblich genutzt werden*).
Alle Schaltungen und technischen Angaben in diesem Buch wurden vom Autor mit größter Sorgfalt erarbeitet bzw. zusammengestellt und unter Einschaltung wirksamer Kontrollmaßnahmen reproduziert. Trotzdem sind Fehler nicht ganz auszuschließen. Der Verlag sieht sich deshalb gezwungen, darauf hinzuweisen, daß er weder eine Garantie noch die juristische Verantwortung oder irgendeine Haftung für Folgen, die auf fehlerhafte Angaben zurückgehen, übernehmen kann. Für die Mitteilung eventueller Fehler sind Verlag und Autor jederzeit dankbar.

*) Bei gewerblicher Nutzung ist vorher die Genehmigung des möglichen Lizenzinhabers einzuholen.

Inhalt

1	Mikroprozessor? – Mikrocomputer?	9
2	Bausteine eines Computers	11
3	Daten – Bits – Bytes	13
4	Encoder – Decoder	17
5	Wie rechnet ein Rechner?	18
6	Was tut ein Programmierer?	20
7	Vereinbarungen	20
8	Zum Aufwärmen – der erste Schritt zum Programmierer	22
9	Der Computer als Rechenmaschine	26
10	Der Computer als Schreibmaschine	30
11	Was ein Taschenrechner nicht kann	34
12	Satzzeichen	36
13	Zurück! Wir rechnen weiter	38
14	Endlich: Wir schreiben unser erstes Programm	42
15	Sie treffen Vereinbarungen mit Ihrem Computer	48
16	Sie bringen Ihren Computer auf Trab	52
17	Nun stellt der Computer die Fragen	62
18	Sie zeigen Ihrem Computer wo's lang geht	64
19	Sie stellen dem Computer Bedingungen	66
20	Zusatzvereinbarungen	68
21	Der Computer als Entertainer	69
22	Textaufgaben	73
23	Der freie Fall	79
24	Zwischenexamen	82
25	Wird jetzt die Mathematik verbogen?	87
26	Jetzt geht's rund!	95
27	Der Computer als Digitaluhr	102
28	Der Computer liest Daten	105
29	Der Computer liest immer noch Daten	111
30	Was ist ein Unterprogramm	117
31	Lauter Zufälle!	120
32	Der Computer als Kaufmannsgehilfe	124
32.1	Die Mehrwertsteuer wird erhöht	124
32.2	Sie beraten Ihre Kunden	126
33	Der Computer als Management-Berater	131
33.1	Wann sind Sie pleite?	131
33.2	Kampf um Marktanteile	135
33.3	Sie planen mit dem Computer Ihre Investitionen	136
34	Der Computer als Textautomat	141

35 Lagertechnik ... 149
35.1 Der Computer verwaltet Ihr Scheckbuch ... 154
35.2 Immer noch: Lagertechnik ... 158
35.3 Der Computer als Telefonverzeichnis ... 160
35.4 Der Computer als Lagerverwalter ... 163
36 Die Logik der Steuerungstechnik ... 171
37 Der Computer als Vermögensberater ... 177
37.1 Sie legen Ihr Geld an! ... 178
37.2 Sie sehen Ihr Kapital wachsen ... 180
37.3 Der Computer verwaltet Ihre Schulden ... 182
37.4 Was kostet Ihr Auto? ... 187
38 Der Computer als Sortiermaschine ... 190
39 Der Computer zum Spielen ... 200
39.1 Das Zahlen-Ratespiel ... 200
39.2 Der Computer bringt Ihre Party in Schwung ... 202
39.3 Sie spielen gegen Ihren Computer ... 206
39.4 Sie erweitern den Tabulator Ihres Computers ... 218
39.5 Noch einmal: Lottozahlen ... 220
39.6 Sind Sie ein Sonntagskind ... 223
Anhang ... 224
Liste der in diesem Buch verwendeten BASIC-Befehle ... 226
Literatur ... 230
Stichwortverzeichnis ... 231

1 Mikroprozessor? – Mikrocomputer?

Diese bereits in der Einleitung benutzten Begriffe sollen zunächst in ihrer Bedeutung erklärt werden; und zwar in der Reihenfolge

M I K R O – C O M P U T E R – P R O Z E S S O R

Nun, wie Sie wissen, gibt es (Groß-)Computer, um die man ganze Hallen baut. Daneben Anlagen, wie sie beispielsweise ein mittlerer Handwerksbetrieb einsetzt. Diese würde man Klein- oder Minicomputer nennen. Und ein

„MIKRO-"

ist ein Kleinstcomputer, wie Sie ihn – vielleicht – vor sich haben. Daß der „Mikro-" der bislang letzte Schritt in der Entwicklung ist, hängt mit der verwendeten Technologie zusammen, auf die wir hier nicht eingehen wollen.
„Mini" oder „Mikro" beschreiben also die räumliche Größe der Anlagen und ihre Leistungsfähigkeit. Die Art und Weise, wie sie funktionieren, ist dabei im Prinzip bei allen gleich.

„To compute" ist ein Begriff aus dem Englischen; zu Deutsch heißt das „berechnen". Damit ist ein

COMPUTER

ein Ding, mit dessen Hilfe man etwas berechnen kann. Bitte achten Sie genau auf die gewählte Formulierung: Es ist bewußt NICHT gesagt, daß ein Computer rechnen kann!!

Wie dem auch sei: irgend etwas wird dabei in dem Computer ablaufen; irgend etwas wird dabei passieren.

Die „Aufsicht" darüber hat eine Steuereinheit, die gewissermaßen als „Befehlszentrale" alle internen Vorgänge in dem Computer steuert. Und diese zentrale Steuereinheit heißt bei Mikro-Computern

M I K R O P R O Z E S S O R .

Andere Begriffe dafür sind: Zentraleinheit

CPU = Central Processing Unit

Klären wir in diesem Zusammenhang gleich noch zwei weitere Begriffe, die uns im folgenden noch begegnen werden: Gerade bei größeren Computern spricht man auch von Datenverarbeitungsanlagen (DVA) – Anlagen also, die

D A T E N V E R A R B E I T E N .

Daten sind dabei Zahlen wie 3, 1/3, 17,5 usw.; aber auch Worte wie MEIER, MUENCHEN; ferner Kombinationen daraus, wie PORSCHESTR 17 oder 17,35 DM.

Und DATEN VERARBEITEN kann bedeuten:

- mit Zahlen rechnen
- Daten sortieren (wie eine Liste von Namen in alphabetischer Reihenfolge aufstellen)
- Daten vergleichen (wie etwa prüfen, ob ein „Datum" mit einem anderen identisch ist)
- Daten codieren (etwa eine Folge von (Schrift)-Zeichen in Zahlen übertragen) usw.

2 Bausteine eines Computers

Alle Tätigkeiten wie Rechnen – Sortieren – Vergleichen usw. können auch von Ihnen, vom MENSCHEN, ausgeführt werden. Zur weiteren Vertiefung der Begriffe soll deshalb mit aller Vorsicht und Zurückhaltung einmal anhand der *Abb. 1* ein Vergleich Mensch – Computer gewagt werden.

Sie lesen oder hören mit Ihren Sinnesorganen. Dem Computer werden die Daten über Eingabegeräte mitgeteilt. Ein Eingabegerät kann eine (Rechen- oder Schreibmaschinen-)Tastatur sein – oder ein Leser für Lochstreifen oder -karten.

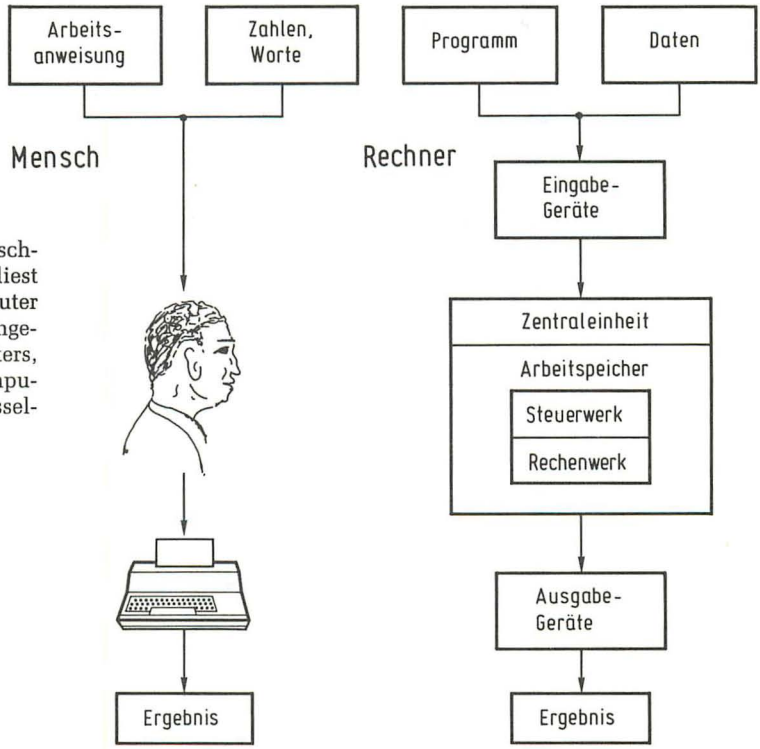


Abb. 1 Vergleich Mensch-Computer. Der Mensch liest oder hört. Dem Computer werden die Daten eingegeben (Aus: M. F. Wolters, Der Schlüssel zum Computer, Econ-Verlag, Düsseldorf)

Mensch
Die Daten werden vom Menschen in sein Gehirn aufgenommen und dort „verarbeitet"

Computer
Die Daten gelangen in die Zentraleinheit des Computers und werden dort „verarbeitet".

Die Rolle des Gehirns beim Menschen spielt also die Zentraleinheit beim Computer.

Das Ergebnis Ihrer Verarbeitung geben Sie Ihrer Umwelt bekannt — Sie reden darüber oder schreiben es auf.

Als Ergebnis der Verarbeitung werden vom Computer Daten ausgegeben.

So wie Sie über Schreibgeräte verfügen, hat der Computer Ausgabegeräte. Das kann ein Bildschirm-Sichtgerät sein, ein Drucker oder ein (Lochstreifen- oder Karten-)Stanzer.

Gehen wir etwas ins Detail:
Wenn Sie rechnen, Daten (Zahlen) verarbeiten, dann tun Sie das nach Anweisungen, nach (Rechen-)Regeln, die Sie irgendwann gelernt haben oder die man Ihnen für die jeweilige Aufgabe mitteilt.

Der Computer verarbeitet Daten nach einem Programm, das Sie (oder der Programmierer) ihm eingeben müssen.

Ebensowenig wie Sie Zahlen sinnvoll miteinander verknüpfen können, wenn Sie die Regeln nicht kennen, ist der Computer in der Lage, mit Daten *allein* etwas anzufangen.
Der Computer verarbeitet die Daten nach einem Programm, das ihm vorschreibt, was er wie mit den Daten anfangen soll. Und hier beginnt unser Beispiel zu hinken. Während Sie zu den Daten, zu den gelernten oder mitgeteilten Regeln noch frei Ihre Erfahrungen, Ihre Phantasie, Ihre Kreativität einbringen können, um zu einem Ergebnis (möglicherweise zu einem falschen!) zu gelangen, kann der Computer *nur* nach dem eingegebenen Programm vorgehen. Ist das lückenhaft oder falsch, wird das Ergebnis zwangsläufig falsch.

3 Daten – Bits – Bytes

Sie haben erfahren, daß 3, 1/3, 17,5,
Meier, München,
Porschestr. 17 und
17,35 DM
Daten sind, die ein Computer verarbeiten kann. Sie werden nicht annehmen, daß man die Daten in der obigen Form in den Computer eingeben kann – man muß sie „computergerecht" aufbereiten. Wenn Sie mit dem Computer verkehren – kommunizieren – wollen, müssen Sie sich einer Sprache bedienen, die beide Partner – Sie und der Computer – verstehen und akzeptieren.

Die *Sprache* ist eine Form der Kommunikation; die *Schrift* eine andere. Leicht fallen Ihnen noch andere Formen ein: die Indianer benutzen Rauchzeichen, die Seeleute Licht- oder Flaggenzeichen, eine Sekretärin verwendet Kurzschriftzeichen, usw.

Wichtig ist: Sie müssen den Zeichen bestimmte Bedeutungen beimessen, die Sender und Empfänger gleichermaßen deuten und verstehen können.

Um es kurz zu machen: Sie verwenden einen bestimmten Code. Es gibt da eine Unzahl von Zuordnungen; wählen wir dafür zwei extreme Beispiele:

Die Chinesen ordnen jedem Begriff ein bestimmtes Symbol (Schriftzeichen) zu; sie kennen einen Vorrat von ca. 40 000 verschiedenen Zeichen und benötigen ca. 3500, um sich in der Zeitung über Alltäglichkeiten zu informieren. Es ist einleuchtend, daß etwas Derartiges für einen Computer in Europa wohl nicht in Frage kommt.

Das andere Extrem:

Für Ihren Code verwenden Sie nur zwei Zeichen – etwa 0 und 1. Den Inhalt einer Information bestimmen Sie durch eine trickreiche Aneinanderreihung nur dieser beiden Zeichen.

Daß man damit – möglicherweise umständlich – alles ausdrücken kann, sehen Sie an einem einfachen Beispiel:

O T T O ,
T O T O .

Hier verwenden Sie auch nur zwei Zeichen, trotzdem haben die beiden daraus gebildeten Worte eine sehr unterschiedliche Bedeutung. Diese auf ZWEI beruhende Form der Codierung ist nun für Computer wie geschaffen, lassen sich doch die beiden Code-Elemente 0 und 1 sehr leicht in elektrischer Form nachbilden.

0	1
Licht aus	Licht an
Es fließt kein Strom...	Es fließt Strom
Ein Schalter ist offen...	Ein Schalter ist zu...
usw.	

Versuchen Sie also, das DEZIMALSYSTEM – das die Ziffern 0...9 verwendet – in ein DUALSYSTEM umzuwandeln, das mit den Zeichen 0 und 1 auskommt!
Der Anfang ist einfach:

DEZIMALSYSTEM	DUALSYSTEM
0	0
1	1

Damit haben Sie sich zunächst festgefahren – eine weitere Unterscheidung ist mit einer Stelle nicht mehr möglich.

In dieser einen Binärstelle läßt sich die kleinste Informations- oder Nachrichtenmenge unterbringen; man nennt sie ein Bit (Bit = Binary Digit).

Wenn Sie auf der dualen Seite eine weitere Stelle spendieren, lassen sich dann zwei, also doppelt so viel weitere Unterscheidungen treffen.
Setzen Sie für

2	10
3	11

Sinngemäß geht es weiter:

4	100
5	101
6	110
7	111
8	1000
9	1001

Damit sind alle 10 im Dezimalsystem benutzten Ziffern in eindeutig voneinander unterscheidbare Dualzahlen umgewandelt. Eines wird dabei deutlich: im Dualsystem benötigen Sie weniger Zeichen, dafür aber mehr Stellen!
Das Ergebnis nochmal in einer Tabelle:

Dezimalzahl	Binärer Code
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101

6	0110
7	0111
8	1000
9	1001

Bevor Sie sich an die Betrachtung mehrstelliger Dezimalzahlen herantrauen, sei einmal detailliert beschrieben, was Ihnen im Laufe Ihres Lebens geläufig geworden ist. Schauen Sie auf die Zahl

1981.

Sie besteht aus 4 Stellen, und durch die bloße Stellung, beispielsweise der Ziffer 9 in dieser Zahl, wissen Sie, daß 9 hier den WERT 900 hat. Im Dezimalsystem verstehen Sie also 1981 wie folgt:

1 mal Tausend
+ 9 mal Hundert
+ 8 mal Zehn
+ 1 mal Eins.

Schreiben Sie Tausend, Hundert usw. als Potenz von 10, dann sieht das so aus:

1 mal 10^3 — denn 1000 ist 10 x 10 x 10 = 10^3
+ 9 mal 10^2 — denn 100 ist 10 x 10 = 10^2
+ 8 mal 10^1 — denn 10 ist 1 x 10 = 10^1
+ 1 mal 10^0 — denn 1 ist 10^0.

Im Dualsystem – um das es bei Computern geht – stellen sich die Zahlen als Potenzen von 2 dar.

1981 würde sich danach wie folgt aufbauen:

	1	mal 2^{10} =	1024
+	1	mal 2^9 =	512
+	1	mal 2^8 =	256
+	1	mal 2^7 =	128
+	0	mal 2^6 =	0
+	1	mal 2^5 =	32
+	1	mal 2^4 =	16
+	1	mal 2^3 =	8
+	1	mal 2^2 =	4
+	0	mal 2^1 =	0
+	1	mal 2^0 =	1
			1981

In einer Dualzahl dargestellt, sieht das so aus:

1 9 8 1 = 11110111101

Fragen Sie nicht, nach welchen Regeln das zustande gekommen ist (selbstverständlich gibt es für das Umwandeln von Dezimal- in Dualzahlen und zurück feste Regeln!). Es ist jedenfalls reichlich unübersichtlich, aus der reinen Binär- auf die Dezimalzahl zu schließen und umgekehrt.

Etwas einfacher und übersichtlicher wird die Geschichte, wenn wir die Dezimalzahl nicht als Ganzes, sondern stellenweise wie folgt codieren:

1	9	8	1	Dezimalzahl
10^3	10^2	10^1	10^0	
0 0 0 1	1 0 0 1	1 0 0 0	0 0 0 1	BCD-Darstellung

Sie erkennen, daß jetzt jede Stelle der Dezimalzahl nach der Übersetzungstabelle auf Seite 14, unabhängig von den anderen Stellen, in eine Binärzahl umgewandelt wurde.

Dieser Code ist der sogen. BCD-Code (BCD = Binär Codierte Dezimalziffer). Und für den gibt es Regeln der Umwandlung; Regeln, wie man mit diesem Code rechnen, wie man darin außer Ziffern auch Zeichen (Buchstaben, Operationszeichen) darstellen kann usw.

Begnügen Sie sich mit diesen Ausführungen und fassen Sie zusammen:

- Weil es technisch besonders einfach ist, (nur) Z W E I Zustände elektrisch nachzubilden, werden in der Computertechnik Ziffern und Zeichen in einem C O D E dargestellt, der nur zwei verschiedene Zeichen kennt. Das einfachste Codewort ist ein *Bit*.
- Nach Regeln und Vereinbarungen, die den Ihnen geläufigen Rechenregeln vergleichbar sind, kann man mit den binär dargestellten Zeichen rechnen oder – allgemein ausgedrückt – D A T E N V E R A R B E I T E N .
- Aus praktischen Gründen faßt man eine bestimmte Anzahl Bit zu DATENWORTEN zusammen; diese haben bei größeren Anlagen 16 Bit.
  Bei Mikrocomputern ist ein Datenwort 8 Bit lang. Man nennt es dann ein Byte.

4 Encoder – Decoder

Zwei zunächst noch widersprüchlich erscheinende Erkenntnisse gilt es noch unter einen Hut zu bringen:

- Sie haben soeben gelernt, daß Computer mit Bits und Bytes gefüttert werden wollen,

ferner

- daß Sie sich beim Umgang mit Computern einer Sprache bedienen müssen, die dieser versteht.

Also können Sie so lange nichts mit Computern anfangen, wie Sie die Bit-Sprache nicht beherrschen?

JAIN.

Wie gehen Sie mit Ihrem Taschenrechner um – der ja die vereinfachte Version eines Computers darstellt? Dort drücken Sie „3 x 3 =" – wie Sie das zu schreiben gewohnt sind und erhalten „9" als ein Ergebnis, das Sie lesen können.

Nun, wenn Sie – beispielsweise – auf die Taste 3 drücken, schließen Sie damit einen Kontakt. Eine Schaltung innerhalb des Rechners (oder Computers) übersetzt das zu 0011 und bietet das der Recheneinheit zur Weiterverarbeitung an. Selbstverständlich gibt der Rechner als Ergebnis seiner Rechnung – beispielsweise – 1001 aus und ein weiterer Übersetzer macht daraus die für Sie verständliche Zahl 9.

Hier werden also automatisch Dezimalzahlen in Dualzahlen codiert, das macht ein ENCODER – und Dualzahlen wieder in Dezimalzahlen (zurück) – codiert, das macht ein DECODER.

5 Wie rechnet ein Rechner?

Bleiben Sie zunächst bei Ihrem Computerzwerg, dem Taschenrechner. Wenn es ein einfaches Gerät ist, das soll hier unterstellt sein – können Sie nur die vier Grundrechenarten +, –, ·, :, durchführen, für die der Rechner bei der Herstellung *fest programmiert* wurde.

In Wirklichkeit können Sie noch weniger. Nämlich nur addieren.
Der Rechner arbeitet:

Addition: $3 + 3 = 6$
Subtraktion: $3 + (-3) = 0$

Durch das Drücken der Subtraktionstaste wird das Vorzeichen des 2. Summanden umgekehrt. Die Addition ergibt dann das richtige Ergebnis 0.

Multiplikation:

Durch Druck auf die Multiplikationstaste wird ein Zähler aktiviert...
Durch den (folgenden) Druck auf die Taste 5 wird der Zähler auf den Anfangswert 5 gesetzt...
Durch eine interne Steuerung zählt der Zähler rückwärts und addiert bei jedem Zählschritt zu dem zuerst eingegebenen Wert 3 den Wert 3 hinzu...
...so lange, bis der Zähler 0 erreicht hat.
Statt 3×5
rechnet der Rechner also

$$3 + 3 + 3 + 3 + 3 = 15$$

Division: $15 : 3$

Hier wird dann sinngemäß abgearbeitet
$15 + (-3) + (-3) + (-3) + (-3) + (-3) = 0$
Der Zähler zählt also, wie oft 3 von 15 abgezogen werden kann, bis das Ergebnis 0 ist. Im Beispiel geht das 5 mal; der Zählerstand 5 wird als Ergebnis der Division ausgegeben.

$$15 : 3 = 5$$

Sicherlich wird Ihr Respekt vor Rechnern und Computern immer geringer, wenn Sie – z. B. – daran denken, was geschieht, wenn der Computer 3 x 1725 rechnen soll.

Dabei sind in Wirklichkeit die einzelnen Schritte eines Rechners oder Computers noch detaillierter; bis zu 1000 einzelner Additions-, Schiebe-, „Speicher rein"-, „Speicher raus"-Operationen sind notwendig für Rechnungen, die Sie lässig im Kopf bewältigen. Das Geheimnis ist die Geschwindigkeit.

Merken Sie sich (für den Umgang mit Computern) den lockeren, aber zutreffenden Spruch:

Wenn man schon dumm ist, sollte man wenigstens fleißig sein...

Und fleißig ist der Computer. Mehr als eine Million mal in der Sekunde kann er die angedeuteten Schritte durchführen, und nur durch die enorme Arbeitsgeschwindigkeit läßt sich mit derartigen Wunderwerken der Technik überhaupt etwas Sinnvolles anfangen.

Das alles geschieht ohne Ihr Zutun durch einen internen Taktgenerator, eine Art innere Uhr, über die der Taschenrechner ebenso verfügt wie der größte Computer.

6 Was tut ein Programmierer?

Nach allem, was Sie bisher wissen, muß es ein ziemlich nervenaufreibender Job sein, den Computer mit Bits und Bytes zu füttern, ihn zu programmieren. Das kann tatsächlich so sein; nämlich dann, wenn es darum geht, mit dem Computer ausschließlich in der Sprache zu sprechen, die er versteht: in der MASCHINENSPRACHE.

Aber die war selbst den Programmierern zu aufreibend, weshalb man im Laufe der Zeit einige höhere Sprachen entwickelt hat, die das Programmieren erleichtern. Dabei haben sich im Laufe der Zeit unterschiedliche Sprachen herausgebildet, je nachdem, ob sie mehr vom Hersteller (des Computers) oder vom Anwender (Hochschule, Industriebetrieb) beeinflußt waren. So gibt es heute beispielsweise

F O R T R A N (FORmula TRANslator, eine 1954 von IBM entwickelte Sprache);
C O B O L (COmmon Business Orientated Language, hauptsächlich für die kommerzielle Datenverarbeitung entwickelt);
A L G O L (ALGOrithmic Language, algorithmische Sprache, vorwiegend für theoretische Zwecke entworfen)

usw.

Für Sie von besonderem Interesse ist

B A S I C (Beginners All Purpose Symbolic Instruction Code, eine leicht zu lernende Sprache für den Anfänger).

Daneben gibt es gewissermaßen noch Dialekte zu den einzelnen Sprachen, die Abweichungen manchmal nur geringfügiger Art darstellen.

Fairerweise sei hinzugefügt, daß ein Programmierer natürlich mehr (und Anspruchsvolleres) tut, als den Computer in seiner Sprache anzureden. Seine wichtigste Aufgabe besteht darin, das per Computer zu lösende Problem zu analysieren und in Plänen, Diagrammen usw. so weit aufzubereiten, daß es einer Programmierung für den Computer zugänglich wird.

7 Vereinbarungen

Damit Sie so rasch wie möglich mit Basic arbeiten können, wollen wir ein paar Verabredungen zur weiteren Vorgehensweise treffen. Um sich Mißerfolge oder Enttäuschungen zu ersparen, sollten Sie wenigstens eine Weile meinen Vorschlä-

gen wortgetreu folgen. Sind Sie erst mit Ihrem Gerät und den ersten Basic-Befehlen vertraut, steht Ihrer Kreativität nichts mehr im Wege.

Ich werde die Computer-Fachsprache vermeiden. Ob etwas ein „Kommando" oder ein „Statement" oder Sonstwas ist, sollte Ihnen (noch) gleichgültig sein. Ich werde immer dann von „Befehlen" reden, wenn Sie – um ein Ergebnis zu erhalten – Ihrer Maschine bestimmte Mitteilungen machen müssen.

Um für Sie die Übersichtlichkeit zu erhöhen, wird dieses Buch vom nächsten Kapitel an „zweiseitig" – das heißt:

Auf der *linken Seite* schreibe ich hin, was Sie tun oder in Ihren Computer eingeben sollen.

Ebenfalls auf der linken Seite,

aber eingerahmt wie hier

finden Sie, was Ihr Computer ausgibt, sobald Sie den Befehl dazu gegeben haben.

Auf der *rechten Seite* werde ich Ihnen die verwendeten Befehle erläutern und weitere Kommentare anbringen.

Sobald ein Befehl zum ersten Mal auftaucht, erscheint er groß und deutlich am Kopf der rechten Seite, so daß Sie auch bei raschem Durchblättern des Buches einen Überblick haben. Außerdem finden Sie im Anhang eine Liste aller in diesem Buch verwendeten Befehle mit der Angabe, auf welcher Seite diese zum ersten Mal aufgetaucht und erklärt sind.

Alle Programme habe ich selbstverständlich ausprobiert – und zwar auf dem TRS-80 von Radio-Shack bzw. auf dem PC 100 von Siemens. Das hat sich zufällig so ergeben und soll nichts über diese oder andere Maschinen aussagen. Haben Sie irgendein Problem, die Programme auf Ihrem Gerät zum Laufen zu bringen – was ich mir nicht vorstellen kann – dann ziehen Sie das Handbuch Ihres Computers zu Rate. Auf deutlich sichtbare Unterschiede weise ich darüber hinaus in einer Tabelle im Anhang hin.

Wir werden gelegentlich eine Weile an einem Programm herumbasteln. Löschen Sie deshalb ein Programm erst, wenn ich Sie dazu auffordere – bzw. dann, wenn ein neuer Übungsabschnitt beginnt.

8 Zum Anwärmen – der erste Schritt zum Programmierer

Und nun schalten wir endlich ein! Nach der von elektrischen Geräten bekannten Anwärmphase sehen Sie links oben etwas auf dem Bildschirm. Bei meinem TRS-80 erscheint:

```
MEMORY SIZE?–
```

Drücken Sie die Taste

[ENTER] [ ] *)

```
MEMORY SIZE?
RADIO SHACK LEVEL II BASIC
READY
>—
```

*) Siehe ①, Seite 224

Damit fragt das Gerät nach der Größe des (noch) zur Verfügung stehenden Speichers. *Das ist für Sie hier noch nicht von Interesse.*

ENTER bewirkt den Abbruch der laufenden und Anfang einer neuen Zeile. Diese Taste entspricht der Taste „Wagenrücklauf" bei einer Schreibmaschine. Die weiteren, möglichen Auswirkungen dieser Taste erfahren Sie bei Bedarf!

RADIO SHACK ist der Hersteller des Gerätes
LEVEL II ist ein Dialekt der Programmiersprache BASIC
READY – das Gerät meldet sich arbeitsbereit.

Der „kleine Strich" geht bei jedem Anschlag um eine (Buchstaben-)Position nach rechts.

Das ist der CURSOR (Zeiger), der Ihnen anzeigt, auf welche Position beim nächsten Tastendruck das gewünschte Zeichen (Buchstabe, Ziffer oder Zeichen) gesetzt wird.

Ich darf annehmen, daß Sie schon einmal einen Taschenrechner in der Hand gehabt und damit gerechnet haben? Gut.

Ich will nicht annehmen, daß Sie schon einmal mit einer (elektrischen) Schreibmaschine geschrieben haben.

Nun sitzen Sie vor Ihrem „Personal-Computer", dessen Tastatur stark an die einer Schreibmaschine erinnert. Einige Modelle – darunter auch mein TRS-80 – haben rechts von der Schreibmaschinentastatur noch einen Tastenblock mit Ziffern und Zeichen.

Ihr erstes Problem: Wenn Sie Zahlen eingeben wollen – welche Tastatur benutzen Sie? Die auf der oberen Reihe der Schreibmaschinentastatur oder die des rechten Tastenblocks? Nehmen Sie, welche Sie wollen!

Entschließen Sie sich zur Verwendung der großen Tastatur – wie ich das aus Gewohnheit tue – dann haben Sie das nächste Problem:

Drücken Sie auf die Taste [% 5] oder [& 6], erhalten Sie die Eingabe 5 bzw. 6.

Und wenn Sie das Zeichen am oberen Rand dieser beiden oder anderer, doppelt belegter Tasten eingeben wollen, also beispielsweise % oder &? Auf der unteren Tastenreihe finden Sie links und rechts je eine doppelt große Taste:

[SHIFT] und [% 5] gleichzeitig gedrückt, ergibt die Eingabe %

Und so weiter..

Wie vereinbart, haben Sie sicher alles mitgeübt und nun eine Menge Unsinn auf dem Schirm. Man sollte halt die Tafel abwischen können!

Ich verrate Ihnen, wie das geht. Drücken Sie die Tasten C L S. Also:

CLS *) danach wieder

[ENTER] [ ]

Sind Sie mit dem Ergebnis zufrieden?

*) Statt die 3 Tasten C-L-S zu drücken, können Sie auch die Taste [CLEAR] [ ] drücken. Was die gleiche Wirkung hat; siehe ⑤, Seite 225

CLEAR SHIFT % 5 = % CLS

Das mit SHIFT geht in Zeitlupe so:

SHIFT drücken – gewünschte Taste drücken – Taste loslassen – SHIFT loslassen.

Das heißt, während der Betätigung der gewünschten Taste sollte SHIFT gedrückt sein.

Wichtig für Ihre weiteren Fortschritte sind folgende Tastenkombinationen:

Addition	SHIFT	+ ;	bedeutet „PLUS“	+
Subtraktion		= –	bedeutet „Minus“	–
Gleichheits-zeichen	SHIFT	= –	bedeutet „Gleich“	=

Für Multiplikation und Division versteht der Computer nur Zeichen, die Sie dafür üblicherweise nicht verwenden:

Multipli-tion	SHIFT	* :	bedeutet „Mal“	*
Division		? /	bedeutet „geteilt durch“	/

CLS heißt „**Cl**ear **S**creen“; „lösche den Schirm!“

Der Bildschirm wird gelöscht, der Cursor erscheint links oben auf der ersten, ausnutzbaren Zeile. Falls ein Programm gespeichert ist – (Sie sind davon noch ca. 30 Minuten entfernt!), wird dieses durch CLS NICHT gelöscht.

Noch etwas zu ENTER In manchen Fällen – wie hier – ist ENTER auch ein Befehl, der die Ausführung des zuletzt Eingegebenen (hier: CLS) direkt bewirkt. Sie lernen darüber noch mehr!

9 Der Computer als Rechenmaschine

Würden Sie jetzt mit Ihrem Taschenrechnerwissen auf Ihren Computer losgehen und – wie beim Taschenrechner – 3+5= eingeben, dann wäre Ihr Computer durch nichts in der Welt zu bewegen, Ihnen das Ergebnis zu verraten.

Der Grund liegt darin, daß Ihr Taschenrechner im Werk fest auf seine Aufgabe programmiert wurde. Ihr Computer aber ist ein frei programmierbarer Rechner – was leider zunächst bedeutet, daß Sie ihm nicht nur die Daten eingeben müssen. Sie müssen ihm auch sagen, was er damit tun soll.

Tippen Sie einmal:

PRINT 3 [SHIFT] [+ ;] 5

[ENTER]

Sobald Sie [ENTER] drücken, zeigt der Bildschirm:

```
MEMORY SIZE?
RADIO SHACK LEVEL II BASIC
READY
  PRINT 3 + 5
  8
READY
>—
```

Sie vermuten richtig: 8 ist das Ergebnis, das der Computer Ihnen anzeigt.

Daß Sie das mit Ihrem Taschenrechner billiger – und mit weniger Manipulationen – auch rauskriegen, war den Vätern Ihres Computers auch klar.

Seien Sie von ihm noch nicht enttäuscht!

PRINT
ist ein Ausführungsbefehl, der – das haben Sie ja auf dem Bildschirm gesehen – *hier* bedeutet: RECHNE und DRUCKE DAS ERGEBNIS AUS!

Wie alle Computerbefehle, so kommt auch PRINT aus der englischen Sprache. Ein „Printer" ist ein Drucker; Print heißt wörtlich DRUCKE!

Wie gesagt, hier, bei Ihrer ersten Rechenaufgabe, bedeutet PRINT Rechne! oder „Schreibe das Ergebnis..."

Für viele Computer gibt es eine nützliche Abkürzung für
PRINT : ?

Geben Sie Ihre erste Aufgabe doch einmal wie folgt ein:

SHIFT [? /] 3 SHIFT [* :] 5

ENTER

Also: ? ist die Abkürzung für PRINT.

Jetzt können Sie einmal alle vier Grundrechenarten durchprobieren! Geben Sie ein:

PRINT 17 [SHIFT] [+ ;] 25
[ENTER]
dann PRINT 24 – 23
[ENTER]
PRINT 5 [SHIFT] [* :] 7
[ENTER]

PRINT 12 [? /] 6
[ENTER]

Als Ergebnis müßten Sie jetzt auf Ihrem Bildschirm stehen haben:

```
MEMORY SIZE?
RADIO SHACK LEVEL II BASIC
READY
> PRINT 17 + 25
  42
READY
> PRINT 24 – 23
  1
READY
> PRINT 5 * 7
  35
READY
> PRINT 12 / 6
  2
READY
> –
```

Haben Sie auch mal ? anstelle von PRINT versucht?

Ich empfehle Ihnen, das eine Weile zu üben, auch wenn's Ihnen noch so lächerlich erscheint. Gehen Sie dabei ruhig etwas „ran"; meine folgenden Hinweise sollen Sie vor Fallstricken bewahren.

- Ob Sie auf einer Schreibmaschine für Null ein kleines o oder ein Großes O oder das richtige Zeichen Ø wählen, Sie erkennen in 1o, 1O oder 1Ø immer „Zehn". Ihr Computer gerät dabei ins Schleudern. Er will korrekt angeboten haben: Null ist Ø !!

- Wenn Sie die Summe von 7 und 3 (nach Adam Riese ist das 10) mit der Summe von 5 und 5 (das ist auch 10) malnehmen wollen, könnten Sie Ihren Computer so füttern: 7 + 3 * 5 + 5.
  Er wird ausgeben: 27. Und das ist falsch! Nach obigem Text muß dabei 100 rauskommen. Sie haben die in den Computer eingebauten Rechenregeln nicht beachtet. Die lauten:
  * wird vor + (oder –) gerechnet
  / wird vor + (oder –) gerechnet
  * und / sind gleichrangig
  + und – sind gleichrangig
  Zu dem Ergebnis 27 ist der Computer gekommen, weil er gerechnet hat: 3*5 = 15 + 7 = 22 + 5 = 27.
  Den obigen Text hätten Sie so eingeben müssen:
  (7 + 3) * (5 + 5).
  Wenn Sie das tun, sagt der Computer 1ØØ.
  Üben Sie diese Klammerei; im weiteren Verlauf des Buches werden wir noch oft darauf zurückkommen.

- Schließlich sollten Sie auch mal rechnen:
  5 mal 3 Komma 5.
  Schreiben Sie das, wie gewohnt, spielt Ihr Computer nicht mit. Für ihn muß es heißen 3.5 (also Punkt statt Komma!).
- Beachten Sie noch, daß der Computer überflüssige Nullen unterdrückt. Und zwar automatisch.

Also		Ø.5	wird	.5
		ØØ.55	wird	.55
		3.5ØØ	wird	3.5
	aber	3.5ØØ1	wird	3.5ØØ1

So, das reicht einstweilen. Je gründlicher Sie das üben, um so schneller erreichen Sie Ihr Ziel. Sie wollten doch Basic lernen?

10 Der Computer als Schreibmaschine

Wenn der Computer schon eine so einladende Schreibmaschinentastatur hat, liegt es doch nahe, ihn einmal als Schreibmaschine zu benutzen.

Ob Sie die Tasten perfekt wie eine Stenotypistin mit allen zehn Fingern bedienen oder das Einfingersystem benutzen, ist dem Computer dabei gleichgültig. Nur treffen müssen Sie halt die richtige Taste!
Tippen Sie einmal ein:

LIEBE TANTE BERTA

[ENTER]

Was der Computer damit anfängt, sehen Sie auf Seite 32, offenbar nichts! Nun kennen Sie doch schon mit PRINT einen Ausführungsbefehl. Versuchen Sie den! Also:

PRINT LIEBE TANTE BETTA ... Ja, so gehts einem Anfänger! BERTA wollten Sie schreiben, nicht BETTA.

Glücklicherweise ist das Korrigieren von Tippfehlern bei einem Computer sehr einfach. Suchen – und finden – Sie die Taste [←] [ ] *)

Wenn Sie da drauf drücken, verschwindet das A; beim nächsten Tastendruck das T, beim dritten schließlich auch das zweite T.

Und nun schreiben Sie noch einmal richtig:
PRINT LIEBE TANTE BERTA

[ENTER]

Das hat dem Computer auch nicht gefallen! Ich will's Ihnen verraten: Tippen Sie
PRINT "LIEBE TANTE BERTA"

[ENTER]

Das Ergebnis Ihrer bisherigen Fingerübungen zeigt der Bildschirm.

*) Siehe ②, Seite 224

Durch jeden Druck auf die Taste ← geht der Cursor um eine Zeichenposition nach links; dabei wird ein etwa dort stehendes Zeichen gelöscht und kann anschließend durch ein anderes überschrieben werden.

Hätten Sie beispielsweise geschrieben LEEBE TANTE BERTA und Ihren Fehler erst am Ende der Zeile bemerkt, dann wäre es mühsam, insgesamt 16mal ← zu drücken. Für solches Mißgeschick hält der Computer eine komfortablere Korrekturmöglichkeit bereit.

Drücken Sie SHIFT ← und die *ganze Zeile* wird gelöscht!

```
MEMORY SIZE?
RADIO SHACK LEVEL II BASIC
READY
> LIEBE TANTE BERTA
? SN ERROR
READY
> PRINT LIEBE TANTE BERTA
  Ø
?SN ERROR
READY
> PRINT"LIEBE TANTE BERTA"
LIEBE TANTE BERTA
READY
> –
```

? SN ERROR ist eine Fehlermeldung des Computers. Er gibt Ihnen damit zu verstehen, daß er Ihre Eingabe nicht begreift. Sie haben gegen die eingebaute Grammatik verstoßen. SN ist eine Abkürzung für Syntax; ? SN ERROR bedeutet sinngemäß „Verstoß gegen die Regeln!"

Bei Ihrem zweiten Versuch hat der Computer zunächst Ø ausgegeben; Er hatte offenbar PRINT wie besprochen als RECHEN-Befehl aufgefaßt; ein Befehl, den Sie ja schon reichlich geübt haben.

Beim letzten Versuch PRINT "LIEBE TANTE BERTA" hat er das gewünschte Ergebnis LIEBE TANTE BERTA ausgegeben; offenbar hat er diesmal PRINT als SCHREIB-Befehl aufgefaßt.

Dabei haben wir nichts anderes gemacht, als den gewünschten Ausdruck in "" („Gänsefüßchen") zu setzen – und genau darauf kommt es an.

PRINT heißt RECHNE und SCHREIBE DAS ERGEBNIS AUF!
PRINT "....." heißt SCHREIBE!

Das wird Ihnen bald in Fleisch und Blut übergehen!
Übrigens: Sie sind noch ca. 15 Minuten vom Programmieren entfernt!

11 Was ein Taschenrechner nicht kann

Nachdem Sie in den ersten Versuchen den Computer als (bescheidene) Rechenmaschine, danach als Schreibmaschine kennengelernt haben, werden Sie neugierig sein, wie man diese Fähigkeiten kombinieren kann. Legen Sie los:

PRINT 3*7 — das war die Rechenmaschine!

[ENTER]

weiter:
PRINT "3*7" — das war die Schreibmaschine!

[ENTER]

und nun:
PRINT "3 * 7 ="; 3 * 7 — ...und das ist die schreibende Rechenmaschine!

[ENTER]

Weil's so schön war, tippen Sie ein (lassen Sie aber kein Zeichen aus!):
PRINT"WIEVIEL IST 3 * 7 ?"; "DREI MAL SIEBEN IST";3 * 7

[ENTER] Und das muß auf dem Bildschirm stehen!

```
MEMORY SIZE?
RADIO SHACK LEVEL II BASIC
READY
> PRINT 3 * 7
  21
READY
> PRINT"3 * 7"
  3*7
READY
> PRINT"3 * 7 =";3 * 7
3 * 7 = 21
READY
> PRINT"WIEVIEL IST 3 * 7 ?"; "DREI MAL SIEBEN IST"; 3 * 7
  WIEVIEL IST 3 * 7 ? DREI MAL SIEBEN IST 21
READY
> –
```

Ich denke, die Beispiele erklären deutlich genug, um was es geht. Sie haben erkannt, daß man in " " ALLES verstecken kann; Ziffern, Buchstaben, Zeichen usw.; immer wird der Computer brav ohne weitere Bearbeitung das einfach hinschreiben, was in " " steht.

Aber es gibt eine Ausnahme! Wenn Sie etwa schreiben:

PRINT "ICH DACHTE,"MEIN SCHWEIN PFEIFT",ALS ICH DAS SAH"

versteht der Computer nur das, was zwischen erstem " und zweitem " steht und wird ausgeben:

```
ICH DACHTE Ø
?SN ERROR
READY
```

Also, wenn Sie schon Ihr Schwein pfeifen lassen wollen oder etwas anderes hervorheben müssen, versuchen Sie es mal so:

PRINT"ICH DACHTE,'MEIN SCHWEIN PFEIFT',ALS ICH DAS SAH"

Wir müssen uns noch mal mit der Eingabe

PRINT"3 * 7=";3 * 7

beschäftigen. Der erste Teil der Zeile ist klar: Sie befehlen dem Computer 3 * 7 = hinzuschreiben.

Aber dann habe ich ein ; (Strich-Punkt oder Semikolon) dazwischen geschmuggelt. Was hat das zu bedeuten?

Nun, dieses Zeichen versteht der Computer als Aufforderung, unmittelbar nach Ausführung des ersten Teils Ihres Befehls das folgende zu tun: PRINT 3*7. Und das ist ein glasklarer Befehl, 3 * 7 auszurechnen – was der Computer, wie Sie auf dem Schirm sehen – auch unverzüglich erledigt.

Beim letzten Beispiel ist ; zweimal verwendet; wenn Sie das Ergebnis betrachten, sind wohl alle Klarheiten beseitigt?

12 Satzzeichen

Sie haben bisher schon eine Menge Zeichen verwendet. Wir sollten diese noch einmal im Zusammenhang durchgehen, damit Ihnen die Verwendung der Zeichen restlos klar wird.

Geben Sie zunächst der Reihe nach das Folgende ein; betrachten Sie danach das Schirmbild und studieren Sie anschließend meine Erklärungen auf der nächsten Seite!

PRINT 3 * 2,5

[ENTER]

PRINT 3 * 2.5

[ENTER]

PRINT "MEIER";"MEIER"

[ENTER]

PRINT "MEIER ";"MEIER"

[ENTER]

PRINT"MEIER","MUELLER","SCHULZE","WEBER"

[ENTER]

Daraus wird:

```
> PRINT 3 *2,5
  6                    5
READY
> PRINT 3 * 2.5
  7.5
READY
> PRINT"MEIER";"MEIER"
  MEIERMEIER
READY
> PRINT"MEIER ";"MEIER"
  MEIER MEIER
READY
> PRINT"MEIER","MUELLER","SCHULZE","WEBER"
  MEIER          MUELLER          SCHULZE          WEBER
READY
> –
```

- In der ersten Zeile wollten Sie wohl rechnen lassen „drei mal zweikommafünf", haben aber aus alter Gewohnheit ein Komma gemacht. Statt des für Ihren Computer vorgeschriebenen Punktes.
  Der Computer hat Ihre Eingabe so verstanden: Drei mal Zwei, *weiterrücken*, 5 – und er hat das auch brav gemacht.
  Was heißt hier *weiterrücken*? Das Komma ist für Ihren Computer ein Tabellierbefehl. Sobald er in einer Zeile auf ein Komma stößt, rückt er auf derselben Zeile eine Spalte weiter und setzt dort seine Arbeit fort. Bei Print ist also der Schirm in (unsichtbare) Spalten eingeteilt.

- Jawohl! 3*2.5 ist richtig eingegeben; der Computer antwortet mit 7.5.

- Den ; hatten wir schon! Hier macht's der Computer nicht besonders schön und klatscht den einen Meier direkt an den anderen.

- Und warum tut er's jetzt nicht? Betrachten Sie einmal genau die Eingabe! Dort ist zwischen dem letzten Buchstaben und " ein Zwischenraum! (Ergibt sich durch einen Druck auf die große Leertaste!) Auch ein Zwischenraum, ein Druck auf die Leertaste oder, wie die Computerleute heute sagen, *ein „Blank" ist für den Computer ein Zeichen!!* Wenn Sie sich angewöhnen, künftig vor dem zweiten " ein Blank zu machen, können Sie solche unschönen Überraschungen vermeiden!

- Hier führe ich Ihnen noch einmal die Funktion des Kommas vor. Sie erkennen daraus, daß mein Computer – und die meisten anderen „Personal-Computer" auch – maximal *4 Spalten* schreiben kann.

Übrigens: Ist Ihnen aufgefallen, daß jetzt auf dem Bildschirm die Zeilen

MEMORY SIZE?
RADIO SHACK LEVEL II BASIC
READY

fehlen? Der Grund: Dieser Computer kann nur 16 Zeilen „schreiben". Ab der 17. Zeile wird die erste nach oben weg-„gerollt".

13 Zurück! Wir rechnen weiter!

Die letzten drei Abschnitte habe ich eingeschoben, um Ihnen etwas Abwechslung zu verschaffen und Sie nicht nur mit der Rechnerei zu nerven. Aber es wäre beschämend, wenn der Computer nicht *mehr* könnte. Er kann's in der Tat!
Es hilft alles nichts – wir müssen noch einmal zurück.
Können Sie sich noch daran erinnern, wie Sie in der Schule die Aufgabe: 2*2*2*2 vereinfacht haben? Erinnern Sie sich an 2^4 (gesprochen: 2 hoch 4)?

Es ist also 2*2 = 4 = 2^2;
2*2*2 = 8 = 2^3;
2*2*2*2 = 16 = 2^4; usw.

Wenn Sie solche Aufgaben mit dem Computer rechnen, ist es dem egal, ob Sie 2*2*2*... eintippen. Aber ich möchte Ihnen eine Vereinfachung beibringen. Unglücklicherweise gibt es für den Computer nicht die Schreibweise 2^4. Aber es gibt einen Umweg:

Tippen Sie: *)

? 2 [↑] [ ] 4

[ENTER]

War's schwer?

Nun zu einer anderen Übung. Sie wissen, daß $\sqrt{16} = 4$ ist (Quadratwurzel aus 16 = 4). Betrachten Sie folgende Tabelle:

$2^2 = 4$; $\sqrt{4} = 2$; $2^3 = 8$; $\sqrt[3]{8} = 2$;
$2^4 = 16$; $\sqrt[4]{16} = 2$;

Auch für $\sqrt{\ }$ hat der Computer kein direktes Zeichen. Für $\sqrt{\ }$ = Quadratwurzel gibt es den Begriff SQR. Tippen Sie ein

? SQR (16)

[ENTER]

Entweder, Sie sind ein so guter Mathematiker, daß Sie das Folgende nachvollziehen können – oder Sie sollten es mir glauben und damit arbeiten. Ich will Ihnen schließlich BASIC beibringen und kein Mathebuch vorsetzen.
Wenn Sie die dritte Wurzel ($\sqrt[3]{\ }$) ziehen wollen, geben Sie ein:

? 8 [↑] [ ] (1/3) (das Ergebnis ist 2); oder $\sqrt[4]{16}$:

? 16 [↑] [ ] (1/4) (das Ergebnis ist ebenfalls 2)

*) Siehe ⑧, Seite 225

- Die Rechnung mit „hoch..." heißt Potenzieren.
  Der Computer benötigt dazu das Zeichen [↑] [ ]

  Beispiel 3^5 wird eingegeben 3 [↑] [ ] 5
- SQR ist eine Abkürzung für **SQ**ARE **R**OOT; zu deutsch „Quadratwurzel".
  Wichtig ist, daß Sie die Zahl, deren Quadratwurzel Sie ziehen wollen, in ( ) setzen.
  Also $\sqrt{4}$ = SQR (4)
- Wollen Sie die dritte, vierte oder ...Wurzel ziehen, dann wählen Sie mangels geeigneter Eingabetasten:

  $\sqrt[3]{...}$ wird [↑] [ ] (1/3)

  $\sqrt[4]{...}$ wird [↑] [ ] (1/4)

  $\sqrt[5]{...}$ wird [↑] [ ] (1/5)

  usw.
- Komplettieren wir nun unsere „Rangliste" von Seite 29

  [↑] [ ] wird vor * und / abgearbeitet

  $\sqrt{}$ wird vor * und / abgearbeitet
  * wird vor + und – abgearbeitet
  / wird vor + und – abgearbeitet

  [↑] und $\sqrt{}$, * und /, + und – sind jeweils gleichwertig.

Ich denke, ein paar Klammerübungen sind noch nützlich! Geben Sie ein:

? 7 + (–3)

ENTER

?((7/2)*(1+4))/1Ø

ENTER

?(5–3)*(1Ø+5))*5

ENTER

?((5–3)*(1Ø+5))*5

ENTER Der Bildschirm zeigt jetzt:

```
MEMORY SIZE ?
RADIO SHACK LEVEL II BASIC
READY
>?7+(–3)
  4
READY
>?((7/2)*(1+4))/1Ø
  1.75
READY
>?(5–3)*(1Ø+5))*5
  3Ø
  ? SN ERROR
READY
>?((5–3)*(1Ø+5))*5
  15Ø
READY
>–
```

- Wenn Sie Geld auf Ihrem Bankkonto haben, dann ist das ein Guthaben; Ihre Bilanz ist *positiv*. Haben Sie dort „Miese" – also Schulden, dann ist Ihre Bilanz *negativ*.
  Der Computer rechnet sowohl mit positiven als auch mit negativen Zahlen.
  Der Computer begnügt sich aber großzügig mit 3, wenn Sie +3 meinen. Bei der Eingabe von positiven Zahlen können Sie auf das Vorzeichen verzichten.
  Bei der Eingabe von negativen Zahlen müssen Sie das Vorzeichen angeben!
  Es gibt Computer, die akzeptieren etwa folgende Eingaben:
  7 + –3 oder 12/–2 oder 5*–7
  Es sieht nicht nur besser aus, sondern wirkt auch übersichtlicher, wenn Sie negative Größen in ( ) setzen, also:
  7+(–3) oder 12/(–2) oder 5*(–7)
- Machen Sie ausgiebigen – und überlegten – Gebrauch von Klammern!
  Merken Sie sich folgendes Schema:

  Der Computer arbeitet zuerst die innere (1) Klammer ab...
  arbeitet danach die zweite (2) Klammer ab...
  arbeitet zuletzt die äußere (3) Klammer ab.
  Achten Sie darauf, daß in einem Ausdruck soviele sich öffnende Klammern „(" enthalten sind, wie sich schließende „)".
  Diese Regel hatten wir bei der vorletzten Aufgabe nicht beachtet. Was den Computer prompt zu der Falschaussage 3Ø – aber immerhin auch zu einer Fehlermeldung – veranlaßte!

14 Endlich! Wir schreiben unser erstes Programm!

Bisher haben Sie Ihrem Computer immer nur einzelne Befehle gegeben. Ein *Programm* ist eine *Liste* von Befehlen, die einer nach dem anderen erledigt werden. Geben Sie ein:

10 PRINT "BERECHNUNG DER KREISFLAECHE"

[ENTER]

20 PRINT "==========
====="

[ENTER]

30 PRINT

[ENTER]

40 PRINT"DER RADIUS IST 25MM"

[ENTER]

50 PRINT"DANN IST DIE KREISFLAECHE";25 [↑] 2*3.16;"QMM"

[ENTER]

Sie haben also ein Programm geschrieben, mit dem Sie die Fläche eines Kreises berechnen wollen, dessen Radius 25 mm ist. Ein solches Programm starten Sie mit einem besonderen Befehl: Tippen Sie ein:

RUN

[ENTER]

Und das ist der Lohn für Ihre Mühe:

```
MEMORY SIZE?
RADIO SHACK LEVEL II BASIC
READY
  10 PRINT"BERECHNUNG DER KREISFLAECHE"
  20 PRINT"= = = = = = = = = = = = = = = = = = = = = = = = =
  =====  "
  30 PRINT
  40 PRINT"DER RADIUS IST 25MM"
  50 PRINT"DANN IST DIE KREISFLAECHE";25 ↑ 2*3.16;"QMM"
  RUN
  BERECHNUNG DER KREISFLAECHE
  = = = = = = = = = = = = = = = = = = = = = = = = = = = =

  DER RADIUS IST 25 MM
  DANN IST DIE KREISFLAECHE 1975 QMM
  READY
  >—
```

Was ist passiert?
Ein Programm beginnt immer mit einer Zeilen-Nummer.
Sie könnten zwar fortlaufend mit 1,2,3,4... numerieren. Aus Gründen, die Sie noch schätzen lernen werden, wählt man aber größere Zeilenabstände. Ich habe mich im Beispiel für 1Ø, 2Ø, 3Ø... entschieden.

Wichtig ist: *Die Befehle werden in der aufsteigenden Reihenfolge der Zeilen-Nummern abgearbeitet.*

In Zeile 1Ø schreiben Sie die „Überschrift".

In Zeile 2Ø unterstrichen Sie diese; Sie wissen ja, daß man in "....." (fast) alles verstecken kann!

In Zeile 3Ø steht nur PRINT. PRINT mit „gar nichts" dahinter veranlaßt den Ausdruck einer Leerzeile.

In Zeile 4Ø geben Sie den Wert 25 für den Radius ein.

In Zeile 5Ø schreiben Sie den „Ausgabetext", gefolgt von der Formel für die Kreisfläche.

Gestartet wird das Programm mit RUN [ENTER]

Hat Ihnen das Ergebnis – der Bildschirm-Ausdruck – gefallen?

Sie machen jetzt Ihre *wichtigste Erfahrung* im Umgang mit Computern! Das Ergebnis sieht zwar hübsch aus. Es ist aber *falsch*! Warum?

Nun, schon seit einiger Zeit gilt für π der Wert 3.1416... – wir aber haben in die Formel, Zeile 5Ø, eingesetzt: 3.16. Und das hat der Computer nicht gemerkt!? Er hat es NICHT!

Bitte merken Sie sich:

- Ein Computer ist schnell, aber dumm!
- Die Intelligenz steckt im Programm, nicht im Apparat!
- Was der Computer ausgibt, kann nicht besser – oder richtiger – sein, als das, was Sie ihm an Daten und Befehlen eingeben!
- Wenn Ihnen jemand erklärt: „Das geht nicht...", „das kann der Computer nicht..." – dann heißt das in Wirklichkeit:
- Das will ich – der Programmierer – nicht! oder
- Das kann ich – der Programmierer – nicht!

Weil bildliche Darstellungen besonders gut haften sollen, noch einmal:

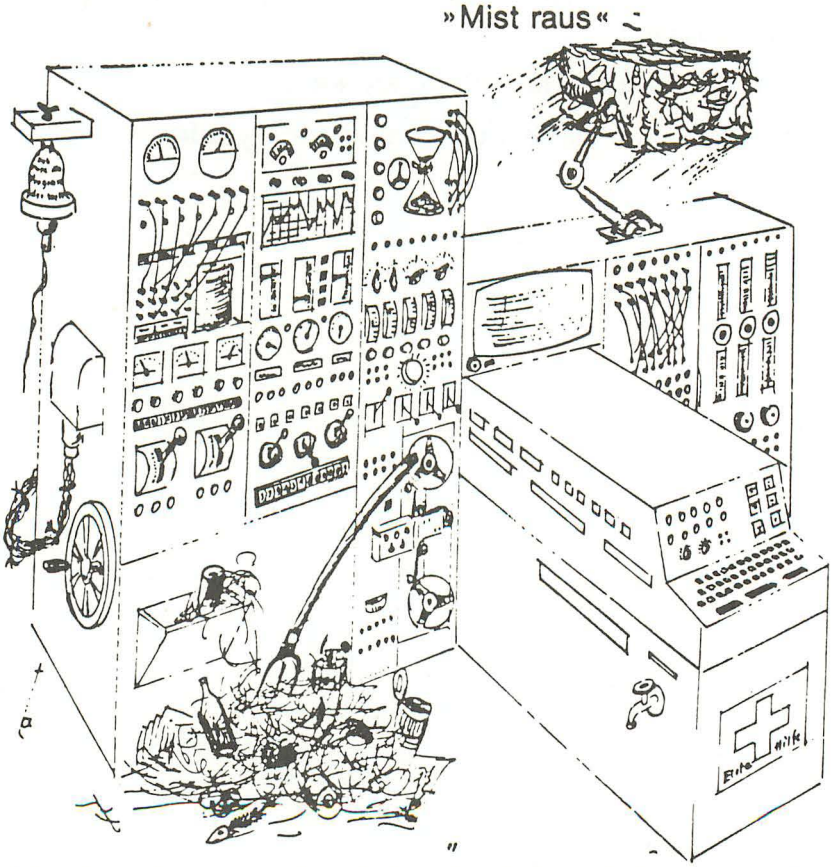


Abb. 2 Auf Englisch: Garbage in, Garbage out! (Aus: M. F. Wolters, Der Schlüssel zum Computer. Econ-Verlag Düsseldorf)

Nach diesem ersten Schock etwas zur Aufmunterung! Ihr letztes Programm steht noch? Wenn nein, geben Sie es bitte noch einmal ein, so wie es war!
Danach geben Sie ein:

50 PRINT"DANN IST DIE KREISFLAECHE";25 ↑ 2*3.1416;"QMM"

Wenn Sie jetzt mit

RUN

[ENTER]

starten, erhalten Sie den gleichen Ausdruck wie vorhin – aber jetzt mit dem richtigen Ergebnis. (Weil Sie jetzt für π den richtigen Wert eingegeben haben.)
Fazit:
Wenn Sie sich geirrt haben, einen Fehler gemacht haben – einen logischen oder einen Tippfehler – dann können Sie das durch Eingabe des richtigen Wortlautes wiedergutmachen.
Sie können in einem Programm einfach den *falschen Zeileninhalt* mit dem *richtigen Zeileninhalt überschreiben.*

Bevor wir uns an das nächste Programm machen, ein praktischer Hinweis:

Angenommen, das letzte Programm ist noch in Ihrem Computer gespeichert. Wenn Sie jetzt ein neues eingeben, könnten sich später bei der Ausführung altes und neues Programm beißen. Es ist zweckmäßig, vor Beginn eines neuen Programms „Klar Schiff" zu machen und ein etwa noch gespeichertes, altes Programm zu löschen. Tippen Sie ein:

NEW

[ENTER]

Geben Sie auf dem nun wieder jungfräulichen Gerät ein:

10 PRINT"WIEDERHOLUNG"

[ENTER]

20 PRINT"DER RADIUS IST"; 10;"MM"

[ENTER]

30 PRINT"DANN IST DER DURCHMESSER"; 2*10;"MM"

[ENTER]

RUN

[ENTER]

Vielleicht stört es Sie, daß jetzt Programm und Ergebnis zusammen auf dem Schirm stehen? Außerdem haben Sie – vielleicht – vergessen, „WIEDERHOLUNG" zu unterstreichen? Lassen Sie alles, wie es ist, und geben Sie ein:

5 CLS

[ENTER]

15 PRINT"==========
====="

[ENTER]

RUN

[ENTER]

Der Mensch ist nie zufrieden. Was tun Sie, wenn Sie an dem trickreich zum Verschwinden gebrachten Programm weiterarbeiten wollen? Geben Sie ein:

LIST

[ENTER]

Nun könnten Sie eigentlich (für das erste) zufrieden sein!

- Sie können ein Programm löschen, wenn Sie einfach das Gerät ausschalten. Weil bei Verlust der Versorgungsspannung die Arbeitsspeicher ihr Gedächtnis verlieren. Für umfangreiche Programme kann das unangenehm sein! Schauen Sie im Handbuch Ihres Computers nach, wie man ein Programm durch Überspielen auf einen Cassettenrecorder „rettet"! (Das lohnt sich aber erst später!).
- NEW löscht das eingegebene Programm. NEW heißt „NEU".
  Hier haben Sie zwei wichtige Erfahrungen gemacht:
- Die nachträglich eingegebenen Zeilen 5... und 15... werden nach RUN vom Computer richtig einsortiert und nach aufsteigenden Nummern abgearbeitet. Da nach Ihrer zweiten Eingabe der erste Befehl 5 CLS lautete, wird auch dieser Befehl als erster ausgeführt. Hier führt das zum Löschen des Bildschirmes.
- Sie erkennen, wie sinnvoll es ist, für die Zeilennummern größere Schritte zu wählen. Nur so ist das nachträgliche Einfügen von Befehlszeilen erst möglich!
- LIST heißt „LISTe auf..., Schreibe nieder..." – nämlich das gespeicherte Programm. Zu diesem Befehl gibt es ein paar nützliche Varianten, die Sie jetzt einmal probieren sollten.

Befehl	bewirkt .....
LIST – 3Ø	Auflisten aller Zeilen von Ø bis 3Ø
LIST 3Ø	Auflisten der Zeile 3Ø
LIST 1Ø – 3Ø	Auflisten der Zeilen 1Ø bis 3Ø
LIST 1Ø –	Auflisten des Programms ab Zeile 10 bis zum Ende des jeweiligen Programms

Übrigens: für – (hier mit der Bedeutung „bis") verwenden Sie die Taste [= –]

15 Sie treffen Vereinbarungen mit Ihrem Computer

Erinnern Sie sich noch an Ihre Schulzeit, an die Anfangsgründe der Mathematik? Dort war bei verschiedenen Aufgabentypen „X" die große Unbekannte, deren Wert es zu ermitteln galt.

Lautet die Aufgabe etwa: X+1=100, dann ist X=100–1, also 99. Heißt es aber X–2=80, dann ist X=80+2, also 82. „X" nennt man hier eine VARIABLE.

Ähnliche Spiele können Sie auch mit Ihrem Computer machen!

Auf Seite 42 hatten Sie bei der Kreisberechnung den Radius mit 25 eingegeben. Wählen Sie in der folgenden Aufgabe den Durchmesser zu 25!

1Ø LET D = 25

[ENTER]

Weiter: Sie wissen, daß der Radius eines Kreises D/2 ist. Also:

2Ø LET R = D/2

[ENTER]

Und Sie wissen, daß die Fläche eines Kreises $R^2 * \pi$ ist. Somit sagen Sie Ihrem Computer:

3Ø LET F = R [↑] [ ] 2*3.1416

[ENTER]

Diese Ihre „ZUWEISUNGEN" merkt sich nun der Computer. Prüfen Sie ihn!

4Ø PRINT D , R , F

[ENTER]

RUN

[ENTER]

```
MEMORY SIZE?
RADIO SHACK LEVEL II BASIC
READY
>1Ø LET D = 25
>2Ø LET R = D/2
>3Ø LET F = R↑2*3.1416
>4Ø PRINT D , R , F
 RUN
 25                    12.5                    49Ø.875
 READY
```

- LET ist ein Zuweisungsbefehl, mit dessen Hilfe Sie Vereinbarungen mit Ihrem Computer treffen können. LET A = 1 könnte man so übersetzen: „A sei 1" oder „Lasse A = 1 sein".
- Damit Sie selbst den Überblick behalten, ist es sinnvoll, Variablen-Namen zu wählen, die einen Zusammenhang mit der Größe haben, die Sie dahinter verbergen wollen. Nicht von ungefähr habe ich in dem Beispiel R gewählt für Radius, D für Durchmesser und F für Fläche.
  Welche Variablen-Namen erlaubt sind, zeigt die Tabelle unten.
- ACHTUNG! Niemals in einem Programm einer Variablen verschiedene Werte zuordnen! (Beispiel: A=5, später dann A=15).
  Der Computer ahndet solche Disziplinlosigkeit damit, daß er (5) vergißt und die letzte Eingabe (15) als verbindlich ansieht!
  (Was wären die Regeln ohne Ausnahmen? – aber davon später!)
- Erlaubt sind als Variablen-Namen

A,B,C, ... Y,Z
AØ,A1 ..... A9
BØ,B1 ..... B9
.....
.....
Z0,Z1 ..... Z9
AA,AB,AC, ... AY,AZ
.....
.....
ZA,ZB,ZC ..... ZY,ZZ

Das sind doch eine Menge Möglichkeiten!

- Variablen-Namen dürfen ein oder zwei Zeichen haben.
  Bei Variablen mit zwei Zeichen muß das erste ein Buchstabe sein!
  Also: A9 ist *erlaubt*; 9A ist *verboten*!
  Fast alle Computer – auch meiner – erlauben folgende Abkürzung:
- Statt LET A9 = 27 einfach eingeben A 9 = 27.
  Probieren Sie, ob das IHR Computer auch toleriert. Ich werde künftig die vereinfachte Schreibweise A = ... wählen!

Weiter im Text!
Lassen Sie das vorige Programm noch stehen!
Außer Zahlenwerten und Ausdrücken (wie R = D/2) können Sie den Variablen beliebige Zeichenketten zuweisen. EGON ist beispielsweise eine Zeichenkette – eine Kette von 4 Zeichen. Wenn Sie eine Zeichenkette einen STRING nennen, gelten Sie schon was in Programmiererkreisen!...

Allerdings müssen Sie den Computer durch ein besonderes Zeichen vorwarnen, daß statt eines Zahlenwertes ein String zugewiesen wird.

- Mit A = 1 weisen Sie der Variablen A den Wert 1 zu...
- Mit A$ = „EGON" weisen Sie der Variablen A$ den Wert EGON zu!

Tippen Sie mal ein:

5 CLS

[ENTER]

8 D$ = "DURCHMESSER"

[ENTER]

25 R$ = "RADIUS"

[ENTER]

35 F$ = "KREISFLAECHE"

[ENTER]

38 PRINT D$, R$, F$

[ENTER]

5Ø END

[ENTER]

RUN

[ENTER]

Und das Ergebnis ist:

```
DURCHMESSER      RADIUS      KREISFLAECHE
25               12.5        49Ø.875
READY
>-
```

Schauen Sie sich doch das Programm noch einmal an! Wie? Mit LIST!

- Ein Dollar-Zeichen ($) an eine Variable angehängt, macht den Computer darauf aufmerksam, daß Sie der Variablen einen String zuordnen wollen.
- Wie Sie das gelernt haben, „verstecken" Sie den String in ".....".
- Sie können allen Variablen der vorigen Liste ein Dollar-Zeichen anhängen. Also von A$ (gesprochen A-Dollar) bis ZZ$ (gesprochen ZZ-Dollar).
- Wie viele Zeichen (+/? Blanks usw. sind AUCH ZEICHEN!!) Sie in einem String unterbringen können, ist bei den einzelnen Computern verschieden. Bei manchen ist die Zahl der Zeichen zwischen "....." begrenzt; z. B. auf 18 Zeichen. Andere Computer sind großzügiger. Der TRS 80 z. B. faßt 256 Zeichen. Probieren Sie das wie folgt aus:

  NEW

  [ENTER]

  1Ø A$ = "... und nun schreiben Sie einige Zeilen ....."

  [ENTER]

  2Ø PRINT A$

  [ENTER]

  RUN

  [ENTER]

  Der Computer druckt nur soviele Zeichen aus, wie er in einem String toleriert. Den Rest läßt er unter den Tisch fallen. Wieviele Zeichen sind das bei Ihnen?
- END: Hier haben Sie in Zeile 5Ø einen Befehl verwendet, der dem Computer sagt END(e; nicht der Fahnenstange, aber des Programms!)
  Der Computer stoppt, sobald er diesen Befehl vorfindet; auch dann, wenn etwa in Zeile 6Ø usw. weitere Befehle stehen sollten.
- Das können Sie sich ruhig angewöhnen: Ihr Programm mit END abschließen!

16 Sie bringen Ihren Computer auf Trab

Eines hat uns der Computer bis jetzt eigentlich noch nicht vorgeführt. Seine affenartige Geschwindigkeit. Und er hat noch nicht bewiesen, daß er uns viel Arbeit abnehmen kann.

Immerhin haben Sie bei der Eingabe mehr geschwitzt als er bei der Ausgabe! Wir wollen den Computer jetzt auf Trab bringen! Geben Sie ein:

NEW

[ENTER]

5 CLS

[ENTER]

1Ø PRINT "RADIUS",
"DURCHMESSER", "FLAECHE"

[ENTER]

2Ø FOR R = 1 TO 1Ø

[ENTER]

3Ø D = 2 * R

[ENTER]

4Ø F = R [↑] [ ] 2 * 3.1416

[ENTER]

5Ø PRINT R , D, F

[ENTER]

6Ø NEXT R

[ENTER]

7Ø END

Und wenn Sie jetzt RUN [ENTER] eingeben, werden Sie verblüfft sein!

RADIUS	DURCHMESSER	FLAECHE
1	2	3.1416
2	4	12.5664
3	6	28.2744
4	8	5Ø.2656
5	1Ø	78.54
6	12	113.Ø89
7	14	153.938
8	16	2Ø1.Ø62
9	18	254.47
1Ø	2Ø	314.16

Da staunen Sie? Lassen Sie das Programm stehen und studieren Sie den weiteren Text!

NEU an diesem Programm sind für Sie die Zeilen 2Ø und 6Ø. Diese beiden Zeilen gehören zusammen; sie bilden eine FOR...NEXT-Schleife.
Damit bringen Sie dem Computer bei, eine Aufgabe mehrmals mit *verschiedenen Werten* für die *gleiche Variable* auszurechnen.

Lassen Sie mich eine Übersetzung versuchen:

FOR	R	=	1	TO	1Ø
für	die Variable R		von 1	bis	1Ø

Die Zeilen 3Ø,4Ø sind Zuweisungen, dann aber

PRINT R , D , F

Das tut der Computer zunächst für die Variable R = 1; das Ergebnis sehen Sie auf der ersten Zeile des Bildschirms. Dann kommt:

NEXT R

Das veranlaßt ihn, zu Zeile 2Ø *zurückzugehen* und das Spiel mit R = 2 zu wiederholen. NEXT R zwingt ihn wieder zurück zu Zeile 2Ø, um mit R = 3 fortzufahren. Dieses Spiel treibt er so lange, bis R = 1Ø erreicht ist. Dies erlaubt ihm, zu Zeile 7Ø zu gehen und sich – weil dort END steht – zur Ruhe zu begeben.

Die FOR...NEXT...-Schleife sollten Sie ruhig ein wenig üben! Vorausgesetzt, Sie sind meiner Empfehlung gefolgt und haben das Programm stehengelassen... (wenn nicht: noch einmal eingeben; Strafe muß sein!)

Dann ersetzen Sie jetzt einmal Zeile 2Ø durch:

2Ø FOR R = 1Ø TO 2Ø

[ENTER]

RUN

[ENTER]

Erlauben Sie mir, auf den Abdruck des Ergebnisses zu verzichten? Fahren Sie fort:

2Ø FOR R = 1 TO 2 STEP .1

[ENTER]

RUN

[ENTER]

Und nun:

2Ø FOR R = 1 TO 1ØØØ

[ENTER]

RUN

[ENTER]

HILFE!!

Drücken Sie [SHIFT] [@] [ ] *)

Was es dazu zu sagen gibt, lesen Sie rechts! Aber das Programm stehenlassen!

*) Siehe ⑥, Seite 225

- Sie können die Variable beliebig mit Werten belegen lassen;
  z. B. ...1 TO 1Ø,...6 TO 23..., 1ØØ TO 115 usw.

- Hier lernen Sie eine Variante kennen.
  FOR R = 1 TO 2 STEP .1
  heißt : „...für R von 1 bis 2 in Schritten von Ø.1..".
  STEP heißt zu deutsch „SCHRITT".
  Sie können beliebige STEPs vorgeben. Geben Sie – wie in dem Beispiel zuvor – NICHTS an, dann wählt der Computer automatisch die Schrittweite 1.
- Naja – man sollte halt nicht so gierig sein!

Mit SHIFT @ [ ] können Sie den Programmablauf „einfrieren".

Bei Druck auf eine beliebige Taste – beispielsweise die breite Leertaste – setzt der Computer seine Arbeit fort.

Immer noch: weiter im Text!
Löschen Sie die Zeilen 3Ø und 4Ø. Wie? Ganz einfach:

3Ø

[ENTER]

4Ø

[ENTER]

Geben Sie ein:

6 D = 2*R

[ENTER]

7 F=R [↑] [ ] 2*3.1416

[ENTER]

2Ø FOR R = 1 TO 1Ø

[ENTER]

RUN

[ENTER]

Was ist jetzt passiert? Der Bildschirm zeigt:

```
          RADIUS      DURCHMESSER     FLAECHE
            1            Ø              Ø
            2            Ø              Ø
            3            Ø              Ø
            4            Ø              Ø
            5            Ø              Ø
            6            Ø              Ø
            7            Ø              Ø
            8            Ø              Ø
            9            Ø              Ø
            1Ø           Ø              Ø
READY
```

- Zeilen-Nummer, gefolgt von [ENTER], löscht die betreffende Zeile aus dem Programm.
  Wollen Sie eine Zeile nicht ersatzlos streichen, sondern ihr nur einen neuen Inhalt geben – der Fachmann nennt diesen Vorgang „Überschreiben" – dann geben Sie ein: Zeilen-Nummer – neuer Zeilen-Inhalt – [ENTER]

- Hier lernen Sie die unerbittliche Logik des Computers kennen. Er tut bekanntlich gar nichts – es sei denn, Sie, sein Programmierer, geben ihm durch klare Anweisungen Gelegenheit zu sinnvollem Tun.

Das alte Programm lautete:	Sie haben das verändert zu:
5...	5...
→ 2Ø FOR R...	6...(Zuweisung)
3Ø...(Zuweisung)	7...(Zuweisung)
4Ø..(Zuweisung)	→ 2Ø FOR R...
5Ø PRINT...	5Ø PRINT...
6Ø NEXT...	6Ø NEXT...

In beiden Programmen beginnt der Computer in Zeile 5 mit CLS.

– geht dann in Zeile 2Ø und „holt" R=1	– erfährt in 6, 7 etwas über D und F, kann damit aber jetzt nichts anfangen
– erfährt in 3Ø, daß D=2*R ist	– geht in 2Ø und holt R=1
– erfährt in 4Ø, daß F=R ↑ 2*3.1416 ist	– kann in 5Ø nur R hinschreiben, weil er nicht weiß, was er (sonst noch) damit machen soll
– führt in 5Ø den „Rechenbefehl" aus	– Sie schicken ihn mutwillig wieder zu 2Ø
– geht über NEXT... zurück zu 2Ø	– dort holt er R=2
– erfährt...usw.	– kann wieder nichts damit anfangen usw.

Blicken Sie ernst auf die obigen Hinweispfeile – und tun Sie es nie wieder!

Leider müssen wir noch einen Augenblick bei der FOR...NEXT...-Schleife bleiben, um eine weitere Fußangel vermeiden zu lernen.

Die FOR...NEXT...-Schleife wird gern dazu benutzt, Tabellen zu zeigen (oder zu drucken). Zu jeder Tabelle gehört bekanntlich ein Tabellen-Kopf, der dem Ganzen nicht nur ein schmuckes Aussehen geben soll, sondern auch deutlich macht, was darunter folgt.

Versuchen Sie doch einmal folgendes Mini-Programm:

```
5 CLS
[ENTER]
10 FOR X = 1 TO 5
[ENTER]
20 PRINT "X =","X * 2 ="
[ENTER]
30 PRINT X , X * 2
[ENTER]
40 NEXT X
[ENTER]
RUN
```

Das Ergebnis ist wirklich nicht berauschend!

```
X =              X * 2 =
  1                2
X =              X * 2 =
  2                4
X =              X * 2 =
  3                6
X =              X * 2 =
  4                8
X =              X * 2 =
  5              10
READY
>—
```

Platz- und Stromverschwendung – so könnte man diese reife Leistung nennen.

Auch hier ist eine ungeschickte, nicht computergerechte Reihenfolge der Befehle gewählt worden.

Sie können ein zufriedenstellendes Ergebnis Ihrer Tabelle erhalten, wenn Sie Ihr Programm wie folgt abändern:

```
10 PRINT "X =","X * 2="
```

[ENTER]

```
15 PRINT"= = = =
```

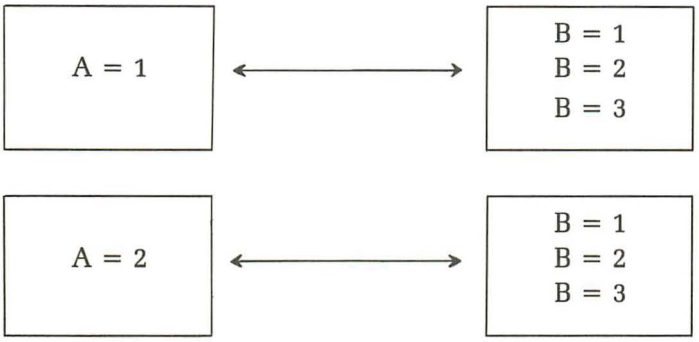
Und immer noch die FOR... NEXT...-Schleife!

Wenn Sie das folgende Programm eintippen, erkennen Sie auf dem Bildschirm, was passiert, wenn Sie in einem Programm zwei FOR...NEXT...-Schleifen anordnen. Um deutlich zu machen, wie das arbeitet, sind auf dem Bildschirm die Variablen A und B mit ausgegeben.

```
NEW
[ENTER]
1Ø FOR A = 1 TO 5
[ENTER]
2Ø FOR B = 1 TO 3
[ENTER]
3Ø PRINT "A= ";A,,,"B= ";B
[ENTER]
4Ø NEXT B
[ENTER]
5Ø NEXT A
[ENTER]
RUN
[ENTER]
```

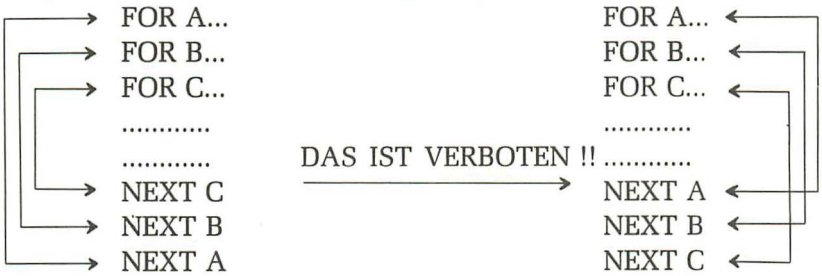
```
A=1          B=1
A=1          B=2
A=1          B=3
A=2          B=1
A=2          B=2
A=2          B=3
A=3          B=1
A=3          B=2
A=3          B=3
A=4          B=1
A=4          B=2
A=4          B=3
A=5          B=1
A=5          B=2
A=5          B=3
```

- Sie erkennen, daß die „B-Schleife" in der „A-Schleife" eingebaut ist.



usw.

- Es können auch mehr als zwei FOR...-NEXT...-Schleifen in einem Programm eingebaut sein. Die korrekte Programmieranweisung lautet:



- Betrachten Sie die Zeile 3Ø! Dort stehen zwischen den beiden Ausdrücken *3 Kommas*!
  - „Komma" bedeutet bekanntlich: „Weiterrücken zur nächsten Spalte"
  - Folgt darauf ein weiteres Komma, wird nochmals zur nächsten Spalte weitergerückt usw.
  - Ergebnis: Der „Ausdruck" erfolgt jetzt in der ersten und vierten Spalte!

17 Nun stellt der Computer die Fragen

Sie lernen jetzt einen überaus nützlichen Befehl kennen – den Befehl INPUT. Dieser Befehl erlaubt es Ihnen, mit Ihrem Computer in ein Gespräch – in einen Dialog – zu kommen. Fangen Sie einmal an:

5 CLS

[ENTER]

1Ø INPUT A

[ENTER]

RUN

[ENTER]

Was sehen Sie auf dem Schirm?

```
1Ø INPUT A
  RUN
?—
```

Hier kommt es auf das FRAGEZEICHEN AN!
Der Computer wartet auf Ihre Eingabe (hier: für A) und gibt das durch ein Fragezeichen zu erkennen. Fahren Sie fort:

1Ø INPUT"GEBEN SIE EINE BELIEBIGE ZAHL EIN !";A

[ENTER]

2Ø INPUT"GEBEN SIE EINE WEITERE ZAHL EIN !";B

[ENTER]

3Ø PRINT"DAS PRODUKT DER ZAHLEN IST :";A*B

[ENTER]

4Ø PRINT"DIE SUMME DER ZAHLEN IST :";A+B

[ENTER]

RUN

[ENTER]

Der Computer wird beginnen mit:
GEBEN SIE EINE BELIEBIGE ZAHL EIN !?
Der Computer wartet jetzt auf Sie!

Tippen Sie:

3

ENTER

Sehen Sie, wie es geht? Tippen Sie ein:

4

ENTER

Und das müßte das Ergebnis sein!

```
GEBEN SIE EINE BELIEBIGE ZAHL EIN !? 3
GEBEN SIE EINE WEITERE ZAHL EIN !? 4
DAS PRODUKT DER ZAHLEN IST : 12
DIE SUMME DER ZAHLEN IST : 7
```

Fassen wir zusammen:

- INPUT...
  veranlaßt den Computer, auf Ihre Eingabe zu warten. Solange diese nicht erfolgt ist – und anschließend ENTER gedrückt wurde – setzt der Computer mit der Abarbeitung des Programms aus.
- Wie aus den Zeilen 1Ø, 2Ø, 3Ø und 4Ø hervorgeht, haben die Befehle PRINT und INPUT eines gemeinsam:
  – Beide Befehle können einen in "....." gesetzten Text ausdrucken.
- Mit INPUT können Sie wie folgt hantieren:

```
1Ø PRINT"GEBEN SIE..."        ODER    1Ø INPUT"GEBEN SIE.....";A
2Ø INPUT A                            2Ø .....
```

  Beachten Sie die Unterschiede des Programms! Die Wirkung auf die Ausführung ist bei beiden Varianten gleich!

18 Sie zeigen Ihrem Computer, wo's lang geht

Das letzte Programm war hoffentlich so niedlich, daß Sie es mehrmals spielen möchten. Lästig ist nur, daß Sie jedesmal mit RUN [ENTER] neu starten müssen.
Ergänzen Sie Ihr Programm um die Zeile:

50 GO TO 10

[ENTER]

RUN

[ENTER]

Merken Sie, wie das geht?
Leider spielt der Computer jetzt dieses Spiel bis zum Erbrechen, und Sie sehen – noch – keine Möglichkeit, ihn davon abzubringen.
Drücken Sie mal auf die Taste:

[BREAK] [ ] *)

Damit bringen Sie ihn offensichtlich zur Vernunft!
Bei der Spielerei eben mag es Sie gestört haben, daß jedesmal mit der Antwort schon die neue, erste Frage auf dem Bildschirm erscheint.
Geben Sie ein:

44 FOR X = 1 TO 1000

[ENTER]

46 NEXT X

[ENTER]

RUN

[ENTER]

*) Siehe ⑦, Seite 225

- GO...TO... veranlaßt den Computer, auf die angegebene Zeile zu springen – statt, wie gewohnt – das Programm zeilenweise abzuarbeiten.
- Als Ziel für den Sprung können Sie eine Zeile angeben, die schon erledigt wurde. So hatten Sie im Beispiel in Zeile 5Ø befohlen, zu Zeile 1Ø zurückzukehren.
- Wenn das Programm es erfordert, können Sie den Computer aber auch auf eine beliebige Zeile nach vorn – etwa von Zeile 5Ø auf Zeile 11Ø – springen lassen.

- Mit [BREAK] [ENTER] unterbrechen Sie die Programm-Ausführung. Die meisten Computer sagen Ihnen dabei durch BREAK IN..., wo sie das Programm abgebrochen haben.
- Wollen Sie das Programm fortsetzen, dann können Sie in diesem Fall zunächst nur *neu starten* mit RUN [ENTER]
- Die FOR...NEXT...-Schleife haben Sie ja schon kennengelernt. Hier führen Sie mit den Zeilen 44 und 46 Ihren Computer buchstäblich an der Nase herum. Sie veranlassen ihn nämlich, 1ØØØmal nach X zu suchen. (Es ist aber gar kein X da, hihi!). Bis der Computer das gemerkt hat – und in Ihrem Programm in Zeile 5Ø fortfährt – *vergeht eine Weile.*

Und auf diese Weile kommt es hier an! Mit diesem Zweizeiler können Sie den Ablauf des Programms an von Ihnen gewählten Stellen beliebig verlangsamen.
Was heißt hier „beliebig"?
Was „beliebig" heißt, erkennen Sie, wenn Sie in Zeile 44 einmal 1ØØØ durch 1ØØ oder 5ØØ oder 1ØØØØ ersetzen!
Keine Erklärungen mehr nötig? Dann weiter!!

19 Sie stellen dem Computer Bedingungen

Den nächsten Befehl stelle ich Ihnen an einem Beispiel vor, mit dem Sie künftig Ihre wertvollen Programme gegen die Benutzung durch Unbefugte sperren können.
Schießen Sie los!

NEW

[ENTER]

5 CLS

[ENTER]

1Ø PRINT"BITTE
KENNWORT EINGEBEN!"

[ENTER]

2Ø INPUT A$

[ENTER]

3Ø IF A$ = "BASIC" THEN 1ØØ

[ENTER]

4Ø PRINT"FALSCHES KENNWORT"

[ENTER]

5Ø PRINT"MIT IHNEN
ARBEITE ICH NICHT!"

[ENTER]

6Ø END

[ENTER]

1ØØ PRINT"ICH STEHE
ZU DIENSTEN, BOSS!"

[ENTER]

11Ø END

[ENTER]

RUN

[ENTER]

Sie haben nun zwei Möglichkeiten:

- Geben Sie auf die Aufforderung des Computers in Zeile 1Ø das hier von mir gewählte Schlüsselwort BASIC ein, dann wird der Computer antworten: ICH STEHE ZU DIENSTEN, BOSS!
- Geben Sie *irgend etwas anderes* – z. B. MEIER – ein, dann antwortet der Computer:

  FALSCHES KENNWORT
  MIT IHNEN ARBEITE ICH NICHT.

  Ist das nicht hübsch?

- Hier kommt es auf die Zeile 30 an: IF...THEN...steht dort; zu deutsch heißt das: WENN...DANN...
  30 IF A$="BASIC" THEN 100
  Das ist bereits Kurzschrift. In Langschrift heißt das:
  IF A$="BASIC" THEN GO TO 100
  Und zu deutsch: WENN (IF!) A$="BASIC",dann (THEN!) gehe zu Zeile 100
- Hier trifft der Computer also eine Entscheidung – die Sie ihm natürlich vorgeben müssen!
- Angenommen...
  ...Sie weisen in Zeile 20 der Variablen A$ den String BASIC zu...
  ...dann „sieht" der Computer in Zeile 30 A$ = BASIC und wird befehlsgemäß zu Zeile 100 gehen...
  ...dort findet er den Befehl, ICH STEHE ZU DIENSTEN,BOSS hinzuschreiben – was er unverzüglich tut.
  ...er geht darauf zur nächsten Zeile 110 und findet dort END – was ihn veranlaßt, Feierabend zu machen.
  (Statt 110 END würde dort natürlich später Ihr Superprogramm starten!)
- Angenommen...
  ...Sie haben in Zeile 20 der Variablen A$ den String MEIER (oder was anderes) zugewiesen...
  ...dann erkennt der Computer in 30, daß MEIER *nicht gleich* BASIC ist...
  (Ja, wirklich, er vergleicht die beiden Strings miteinander!!)
  ...daraufhin sieht er keine Veranlassung, zu Zeile 100 zu gehen, sondern geht – wie er das gelernt hat – in die nächste Zeile 40. Dort steht: FALSCHES KENNWORT – was er hinschreibt, um zur nächsten Zeile zu gehen. Dort steht: MIT IHNEN ARBEITE ICH NICHT. Hat er das hingeschrieben, findet er in der nächsten Zeile 60 END.

Achtung, wichtig:

In Zeile 60 MÜSSEN Sie das Programm mit END abbrechen! Tun Sie das nicht (wollen Sie nicht spaßeshalber mal 60 löschen?), dann würde der auf zeilenweises Abarbeiten dressierte Computer nach Zeile 50 als nächste die Zeile 100 antreffen und den Blödsinn hinschreiben ICH STEHE ZU DIENSTEN,BOSS!

Ich kann es Ihnen nicht oft genug sagen: Der Computer ist NICHT intelligent – allenfalls sind Sie es, sein Programmierer!

Übrigens – statt eines Kenn-„Wortes" hätten Sie auch eine Kenn-„Zahl" verwenden können! Also

...INPUT A
...IF A = 10...

20 Zusatzvereinbarungen

Ich gratuliere Ihnen!
Sie haben wirklich schon das Wichtigste über Basic erfahren. Auf was es jetzt ankommt, ist, das Gelesene solange einzuüben, bis es zu Können wird.

Schon eingangs hatte ich Ihnen ja versprochen, daß Sie lernen werden, eine Handvoll Basic-Befehle virtuos zu beherrschen. Ich halte das für besser – und ich wiederhole mich gern – als mit dem D-Zug durch alles zu sausen, was Basic heißt... Wir werden dazu miteinander Programme erarbeiten, die Ihnen die ganze Vielfalt der Computerei vor Augen führen. Hoffentlich gelingt es mir, Ihre Phantasie anzuregen. Sie könnten dann gemeinsam erarbeitete Programme variieren und Ihre eigenen daraus machen.

Sie werden dabei sehr rasch erkennen, daß das Hantieren mit Basic-Befehlen nicht der wichtigste Teil für den Umgang mit Computern ist. Daß es vielmehr darauf ankommt, Ihre Gedanken zu ordnen, Ihre Probleme zu analysieren und dem Computer alles mundgerecht aufzubereiten. Ich werde versuchen, Ihnen dabei Schützenhilfe zu leisten, damit sich Ihr Schreck über diese Erkenntnis nicht in Lustlosigkeit verwandelt.

Selbstverständlich werden Sie noch eine Reihe von BASIC-Befehlen hinzulernen. Wir werden das bei der Behandlung der einzelnen Programme erledigen. Da Sie jetzt schon einen gehobenen Anfängerstatus haben, werden Sie mir gewiß erlauben, daß ich künftig darauf verzichte, Ihnen jeden Tastendruck vorzuschreiben und jeden Computer-„Ausdruck" in einen Rahmen zu pressen. Zur Belohnung – und zur Entspannung – wollen wir zunächst ein Spiel spielen. Schließlich ist es an der Zeit, daß Sie Ihren Freunden und Verwandten als Computerfachmann imponieren!
Ich wünsche Ihnen weiterhin viel Spaß mit Basic.

21 Der Computer als Entertainer

Sie werden jetzt ein Programm für ein Ratespiel schreiben!
Beginnen Sie mit:

```
5 CLS
1Ø INPUT"GEBEN SIE EINE BELIEBIGE,GANZE ZAHL ZWISCHEN 1 UND 5
EIN..... ICH WERDE DIE ZAHL ERRATEN !";Q
```

Sicherlich sind Sie schon so weit in der Computermaterie drin, daß Sie ahnen: dem Computer muß man nicht nur die Fragen eingeben, sondern auch die Antworten!

Auf die Aufforderung in Zeile 1Ø gibt es – zunächst – 5 mögliche Antworten, nämlich 1,2,3,4 oder 5.

Legen Sie los:

```
2Ø ON Q GO TO 1ØØ,2ØØ,3ØØ ,4ØØ,5ØØ *)
1ØØ PRINT"DIE EINGEGEBENE ZAHL IST 1"
2ØØ PRINT"DIE EINGEGEBENE ZAHL IST 2"
3ØØ PRINT"DIE EINGEGEBENE ZAHL IST 3"
4ØØ PRINT"DIE EINGEGEBENE ZAHL IST 4"
5ØØ PRINT"DIE EINGEGEBENE ZAHL IST 5"
```

Testen Sie einmal diesen Programmabschnitt! Sie werden eine merkwürdige Entdeckung machen:

Beantworten Sie die Frage nach Q mit 5, so wird der Computer die richtige Antwort geben, nämlich: DIE EINGEGEBENE ZAHL IST 5.

Geben Sie aber für Q die Zahl 1 ein, so wird der Computer Ihnen zu Ihrer Enttäuschung den ganzen Katalog von 1 bis 5 servieren. Wenn Sie nicht von selbst darauf kommen, warum er das tun *muß* – so, wie Ihr Programm nun einmal ist – dann lesen Sie noch einmal schnell auf Seite 67 nach! Nicht nötig? Also weiter:

```
15Ø END
25Ø END
35Ø END
45Ø END
55Ø END
```

*) Die Erklärung für diesen – noch – unbekannten Befehl finden Sie am Ende des Programms!

Wenn Sie jetzt das Programm testen, wird der Computer jede eingegebene Zahl, die zwischen 1 und 5 liegt, mit absoluter Sicherheit richtig erraten.

Ihre soeben erreichte Zufriedenheit wird sich in Enttäuschung verwandeln, wenn Sie die Frage nach Q einmal mit 7 oder irgendeiner Zahl beantworten, die NICHT zwischen 1 und 5 liegt! Der so klug erscheinende Computer wird ins Schleudern geraten und falsche Antworten geben!

Der Grund liegt darin, daß Sie sich mit Ihrer Falscheingabe nicht an die mit dem Computer vereinbarten Spielregeln halten. Vereinbart war (in Zeile 1Ø): Es soll eine ganze Zahl eingegeben werden (.5 z. B. ist KEINE ganze Zahl!). Außerdem soll die einzugebende Zahl zwischen 1 und 5 liegen – und das ist bei 7 NICHT der Fall!!

IHNEN traue ich zu, daß Sie sich von nun an an die vereinbarten Regeln halten. Aber Sie wollen doch Ihre Freunde raten lassen! Meine Freunde jedenfalls würden sich einen Spaß daraus machen, einfach mal auf die Taste „A" oder „9" zu drücken! Wäre das eine Blamage!

Es hilft alles nichts: Sie müssen den Computer dazu bringen, solchen Spaßvögeln die gebührende Antwort zu geben!

Anders ausgedrückt:

Sie müssen eine Sperre einbauen gegen alle Eingaben, die nicht den vereinbarten Regeln entsprechen. Sodann müssen Sie dem Computer die Anweisung geben, was er zu tun hat, wenn eine Falscheingabe erfolgt.

Fangen wir damit an:

```
6ØØ PRINT"FALSCH !! DIE EINGEGEBENE ZAHL LIEGT NICHT ZWISCHEN 1
UND 5 ! .....BITTE NACH AUFFORDERUNG NEU EINGEBEN !"
```

Und nun die erwähnte Sperre:

15 IF Q<1 THEN 6ØØ <:das geben Sie so ein: SHIFT [<,]

16 IF Q>5 THEN 6ØØ >:das geben Sie so ein: SHIFT [>.]

Zur Abrundung und Vertiefung basteln Sie noch etwas an Ihrem Programm. Zunächst:

```
6 FOR X = 1 TO 1ØØØ
7 NEXT X
8 CLS
```

Machen Sie nach Eingabe der Zeilen 6, 7 bzw. 8 einen Probelauf, um die Auswirkungen Ihrer Eingaben kennenzulernen!

Als weitere Verzierungen geben Sie ein:

```
11 PRINT
12 PRINT
13 PRINT
```

Nun wäre Ihr Programm eigentlich fertig. Aber es hat noch einen Schönheitsfehler. Wenn Sie ihn selbst abstellen können, hätten Sie schon viel gelernt! Können Sie nicht? Meine Schuld!
Überschreiben Sie Ihr Programm wie folgt:

```
150 GO TO 6
250 GO TO 6
350 GO TO 6
450 GO TO 6
550 GO TO 6
650 GO TO 6
```

Wenn Sie nun neugierig auf Ihr erstes, größeres Programm sind, sehen Sie es sich in Etappen an!

LIST –20; danach LIST 20–. Im Ganzen sieht Ihr Programm dann so aus:

```
5 CLS
6 FOR X = 1 TO 1000
7 NEXT X
8 CLS
10 INPUT"GEBEN SIE EINE BELIEBIGE GANZE ZAHL ZWISCHEN 1 UND 5
EIN...
...ICH WERDE DIE ZAHL ERRATEN !";Q
11 PRINT
12 PRINT
13 PRINT
15 IF Q<1 THEN 600
16 IF Q>5 THEN 600
20 ON Q GO TO 100,200,300,400,500
100 PRINT"DIE EINGEGEBENE ZAHL IST 1"
150 GO TO 6
200 PRINT"DIE EINGEGEBENE ZAHL IST 2"
250 GO TO 6
300 PRINT"DIE EINGEGEBENE ZAHL IST 3"
350 GO TO 6
400 PRINT"DIE EINGEGEBENE ZAHL IST 4"
450 GO TO 6
500 PRINT"DIE EINGEGEBENE ZAHL IST 5"
550 GO TO 6
600 PRINT"FALSCH ! ! DIE EINGEGEBENE ZAHL LIEGT NICHT ZWISCHEN 1
UND 5...
...BITTE NACH AUFFORDERUNG NEU EINGEBEN!"
650 GO TO 6
```

Und was haben Sie dazugelernt?

- Zeile 2Ø: ON Q GO TO ... ist eine Erweiterung des GO...TO...-Befehls.
  Der Befehl bedeutet:
  ON Q = 1 , 2 , 3 , 4 , 5 .....
  GO TO 1ØØ,2ØØ,3ØØ,4ØØ ,5ØØ.....
  Wird also für Q 3 eingegeben, springt der Computer zu Zeile 3ØØ, wird 5 eingegeben, springt er zu Zeile 5ØØ usw.
- Zeilen 1Ø, 6ØØ: Ein Befehl kann auch über zwei Bildschirmzeilen gehen. Sie müssen nur am Ende der ersten — *ohne die Taste* RETURN *zu drücken* – weiterschreiben !!
- Zeile 15: < heißt zu deutsch „kleiner als". Tasten: SHIFT < ,
- Zeile 16: > heißt zu deutsch „größer als". Tasten: SHIFT > .

Ein Beispiel: 3<5; aber 5>3!
Und nun noch als Belohnung eine Anregung zur Abwandlung der Zeile 6ØØ.
Schreiben Sie 6ØØ PRINT"... beliebiges Schimpfwort..."
Ich habe mal geschrieben PRINT"...DU ARMLEUCHTER!"
Sie hätten das Gesicht sehen sollen!

22 Textaufgaben

Haben Sie die in der Schule auch so gern/ungern gemacht? Nun – hier ist eine: *Gegeben* ist die folgende Schaltung aus drei Widerständen (Abb. 3).
Gesucht sind: Der Ersatzwiderstand RE aus R 1 und R 2, der wirksame Gesamtwiderstand RG, der Strom I und der Spannungsabfall U 1 am Widerstand R 3.
Wenn Sie Elektriker sind, werden Sie die Lösung bald haben. Aber wenn Sie keiner sind? Oder es ist schon 20 Jahre her, daß Sie so etwas „durchgenommen" haben?

Dann ist guter Rat (zunächst) teuer. Um es gleich zu sagen: Ihren Computer können Sie für ein paar Minuten vergessen – der hilft Ihnen nicht weiter. Dem müssen Sie nämlich die Aufgabe erst computergerecht servieren, bevor er zur Lösung schreitet. Dieses „Servieren" aber ist keine Frage von Basic, sondern – in diesem Falle – von Elektrotechnik, erstes Lehrjahr. Ich breite das deshalb hier so genüßlich vor Ihnen aus, weil Ihnen so etwas beim Arbeiten mit Ihrem Computer noch oft passieren wird.

Wenn Sie die sachliche Materie nicht beherrschen, nutzt Ihnen auch perfektes Basic gar nichts. Aber es gibt einen Rat für solche Katastrophen: Vielleicht haben Sie einen Freund oder Arbeitskollegen, der Ihnen weiterhelfen kann? Oder Sie besorgen sich ein „Tabellenbuch". Auch ein Fachbuch für den Berufskundeunterricht für das betreffende Sachgebiet leistet gute Dienste. Ich habe davon für einige der noch folgenden Programme auch Gebrauch gemacht.
Zugegeben – die Aufgabe an sich ist simpel. Aber sie führt Ihnen mit voller Wucht vor Augen, auf was es bei der „Computerei" ankommt.
Bevor ich – als Ihr „Arbeitskollege" – Ihnen weiterhelfe, empfehle ich Ihnen die Lektüre der folgenden Merksätze. Sowie tiefes Nachsinnen darüber...
Wenn es nicht so teuer wäre, hätte ich die Merksätze mit einem goldenen Rahmen versehen:

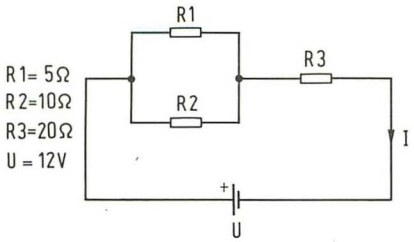


Abb. 3 Ihre erste eingekleidete Aufgabe für den Computer: Berechne den Ersatzwiderstand RE aus R1 und R2, den Gesamt-Widerstand RG, den Strom I durch die Schaltung und den Spannungsabfall an R3

SIE HABEN EIN PROBLEM. (nämlich die Textaufgabe)	DER COMPUTER HAT KEINE PROBLEME.
SIE KENNEN DEN LÖSUNGSWEG.	DER COMPUTER KENNT IHN NICHT.
SIE SIND IN DER LAGE, PROBLEM UND LÖSUNGSWEG IN FÜR DEN COMPUTER VERSTÄNDLICHE BEFEHLE UMZUSETZEN.	DER COMPUTER SCHAUT IHNEN DABEI GELANGWEILT ZU.

Und nun zur Lösung!
Es empfiehlt sich bei solchen Aufgaben, Papier und Bleistift bereitzuhalten, um Skizzen und Zwischenrechnungen zu machen – so, wie ich das hier vor Ihnen tue.

1. Schritt: Sie ermitteln den Ersatzwiderstand RE aus R 1 und R 2 (Abb. 4).
Dafür gibt es die Formel: RE=(R1*R2)/(R1+R2)

2. Schritt: Sie ermitteln den Gesamtwiderstand RG (Abb. 5).
Der ist: RG=RE+R3

3. Schritt: Sie errechnen den Strom I (Abb. 6).
Das gelingt nach: I=U/RG

4. Schritt: Sie machen sich an den Spannungsabfall U1 (Abb. 7).
Dafür gilt: U1=I*R3; für I setzen Sie das in Schritt 3 Entwickelte ein und erhalten U1=(U/RG)*R3.

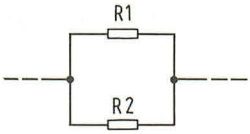


Abb. 4 Welchen Widerstand RE haben zwei parallel geschaltete Widerstände zusammen?

Abb. 5 Welchen Widerstand haben alle drei zusammen?

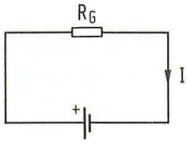


Abb. 6 Wer kennt das Ohmsche Gesetz?

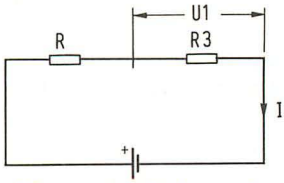


Abb. 7 Wie hoch ist der Spannungsabfall hier?

Und nun können Sie sich wieder Ihrem Computer zuwenden! Sagen Sie dem doch erst einmal, was Sie ganz sicher wissen:

```
10 R1 = 5
20 R2 = 10
30 R3 = 20
40 U = 12
```

Und jetzt vertrauen Sie ihm die so mühsam erarbeiteten Formeln an!

```
50 RE = (R1*R2)/(R1+R2)
60 RG = RE+R3
70 I = U/RG
80 U1 = (U/RG)*R3
```

Der Rest ist jetzt für Sie nur noch Routine!

```
90 PRINT"DER ERSATZWIDERSTAND IST ";RE;" OHM"
100 PRINT"DER GESAMTWIDERSTAND IST ";RG;" OHM"
110 PRINT"DER STROM IST ";I;" AMPERE"
120 PRINT"DER SPANNUNGSABFALL IST ";U1;" VOLT"
```

jetzt noch:

```
5 CLS
```

und los!

Wenn Sie jetzt meckern, daß Sie das mit einem Taschenrechner schneller gemacht hätten – Kenntnis der Formeln vorausgesetzt – dann kann ich Ihnen nur zustimmen. Für eine Einmalaufgabe ist ein Computer sicher ein zu teures Möbel. Aber was ist, wenn Sie diese Aufgabe x-mal mit anderen Werten rechnen müssen? Versöhnen Sie sich mit Ihrem Computer! Variieren Sie einmal – der Reihe nach:

```
40 U = 1
```

danach 100, 220, 1000 usw.!

Oder:

```
40 INPUT"WIE GROSS IST DIE SPANNUNG U ";U
130 FOR X = 1 TO 1000 : NEXT
140 GO TO 5
```

Was haben Sie dazugelernt?

- Genau so logisch, wie Sie die Formeln entwickelt haben, sollten Sie bei der Programmierung zu Werke gehen!

- Der Computer könnte Ihnen – beispielsweise – in Zeile 90 RE nicht ausrechnen, wenn Sie ihm nicht vorher – in Zeilen 10, 20, 50 gesagt hätten, welche Werte er in die Formel einsetzen soll.

Oder:

- Sie könnten billigerweise nicht in Zeile 1ØØ nach RG fragen, wenn Sie nicht vorher in Zeile 5Ø etwas über RE und in Zeilen 1Ø, 2Ø etwas über die für RE nötigen Größen R1 und R2 gesagt hätten!
- Denken Sie deshalb immer daran, daß der Computer nach aufsteigenden Zeilen-Nummern arbeitet! (Es sei denn – wie Sie wissen – Sie befehlen ihm ausdrücklich etwas anderes!)
- Bei dieser Gelegenheit habe ich Ihnen in Zeile 13Ø etwas Neues untergeschmuggelt:
  Der ":" (Doppelpunkt) trennt BASIC-Befehle voneinander. Das heißt, Sie können auf eine Zeile auch mehrere Befehle schreiben (wenn Sie Platz haben!) und wenn Sie die einzelnen Befehle mit ":" voneinander abtrennen!
- Die Ihnen schon geläufige Warteschleife ist auf eine Kurzform gebracht, die man sich leicht merken kann:
  ...FOR X = 1 TO... : NEXT
  Und noch etwas ist neu für Sie:
- In dem obigen Befehl steht nur ... : NEXT
  das sonst gebrauchte ...X habe ich mir geschenkt. Das liegt daran, daß mein Computer „benutzerfreundlich" ist und dies als besonderen Komfort bietet. Sie sollten prüfen, ob Ihrer das auch kann! Wenn nicht, bleibt es bei : ...NEXT X.

Noch eine Textaufgabe...!
Gegeben: Sie haben einen Bleiakkumulator mit 12 V Klemmenspannung. Der Innenwiderstand soll .1 Ohm sein.
Gesucht: Sie sollen die Größe eines Widerstandes R bestimmen, bei dem an der obigen Batterie die maximal mögliche elektrische Leistung P erzeugt wird.
Lösungsweg: Für die Lösung gibt es einen bestimmten Zusammenhang – aber der ist nicht einmal jedem Elektriker auf Anhieb geläufig.
Ein Weg zur Lösung wäre folgender: Sie kaufen sich eine Rolle Widerstandsdraht und probieren...
Auf diese Weise könnten Sie sich dem richtigen Wert für R „empirisch" nähern!
Das müßte doch ein Fressen für schnelle und willige Computer sein!
Hier sehen Sie zunächst einmal das Ersatzschaltbild (Abb. 8).
Und die notwendigen Formeln lauten:

$I = U / (R + R1)$
$P = I^2 * R$

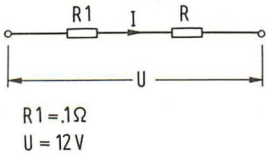


Abb. 8 Welchen Wert soll R annehmen, um maximale Leistung umzusetzen?

Wir entschließen uns, die Lösung über eine FOR...NEXT...-Schleife zu suchen und teilen dem Computer in bewährter Weise erst einmal mit, was wir schon wissen.

```
5 CLS
10 U = 12
20 R1 = .1
30 REM WIDERSTANDSOPTIMIERUNG
40 REM GEGEBEN IST U = 12 V, R 1 = .1 OHM
50 REM GESUCHT IST DIE MAX. LEISTUNG P AN R
60 PRINT"STROM I =" ,"WIDERSTAND R =","LEISTUNG P ="
70 PRINT" = = =
```

Das Leistungsmaximum wird erreicht, wenn der Lastwiderstand R gleich dem Innenwiderstand R 1 ist...

Was haben Sie dazugelernt?

- Die Zeilen 3Ø, 4Ø und 5Ø beginnen mit REM.
  REM ist die Abkürzung von REMark – und das heißt frei übersetzt „Anmerkung".
  Dies ist ein überaus wichtiger Befehl, erlaubt er Ihnen doch, in Ihrem Programm Anmerkungen zu machen oder Erklärungen unterzubringen, die es Ihnen ermöglichen, Ihr Programm auch nach längerer Zeit zu verstehen!
  Durch REM wird der Computer vorgewarnt, daß jetzt etwas kommt, was für Sie – den Programmierer – bestimmt ist und nicht für ihn – Ihren Helfer. Was bedeutet, daß der Computer alles, was auf REM... folgt, *ignoriert*.
  Nach dieser Vorführung von REM werde ich darauf künftig verzichten. Weil ich Ihnen die Programme so ausführlich kommentiere, daß sich die Verwendung von REM wohl erübrigt.
  Aber Sie sollten bei eigenen Programmen davon ausgiebig Gebrauch machen!
- Falls Sie nicht ohne zu überlegen nachvollziehen können, warum ich diesesmal die Formeln erst in den Zeilen 1ØØ und 11Ø untergebracht habe und nicht – wie in der Textaufgabe davor – gleich im Anschluß an die Eingabe der bekannten Größen, dann sollten Sie sich noch einmal die Seiten 52...57 zu Gemüte führen!

23 Der freie Fall

In diesem Programm wenden wir uns zur Abwechslung einmal der Mechanik zu. Aus „Friedrichs Tabellenbuch"*) habe ich mir notiert:
Die Beschleunigung beim freien Fall beträgt G**) = 9.81 m/s².
Die Endgeschwindigkeit des frei fallenden Körpers nach T Sekunden ist
V = G * T.
Die Fallhöhe H ist (G * T²) / 2.
Die Fallzeit T ist $\sqrt{2H/G}$.
Ihre Aufgabe ist es nun, die Endgeschwindigkeit V und die Fallzeit T auszurechnen oder, besser – ausrechnen zu lassen – für „beliebige" Fallhöhen H.

Das Wort „beliebige..." soll Ihre Aufmerksamkeit darauf lenken, daß diese Größe variiert werden soll. In Ihrem Computerprogramm würden Sie das am elegantesten über einen INPUT-Befehl erledigen.

Es geht los:
50 INPUT"GEBEN SIE DIE FALLHOEHE IN 'METER' EIN !";H
Nun geben Sie – nach altem Brauch – das ein, was Sie wissen. Das ist nicht viel – aber immerhin:
60 G = 9.81
Nun habe ich – listig! – zuerst nach der Endgeschwindigkeit V gefragt. Die Formel dafür lautet: V = G * T

Wenn Sie das dem Computer jetzt eingeben, kann er damit nichts anfangen. Er kennt nur G (aus Zeile 60); T ist für ihn noch ein böhmisches Dorf. Da wir eisern konsequent bleiben wollen, geben wir zunächst – computergerecht – ein:
70 T = SQR (2 H / G)
Warum? Weil Sie H über INPUT (Zeile 50) eingegeben haben und G in Zeile 60; der Computer kann also T ermitteln aus Größen, die Sie ihm inzwischen verraten haben. Und nun:
80 V = G * T
90 PRINT T , V
Das Ergebnis wird sein:
? SN ERROR IN 70

*) **Friedrich:** „Tabellenbuch für Elektrotechnik", Ferd. Dümmler Verlag, Bonn.
**) Die Formeln sind – abweichend von der Norm – in Großbuchstaben dargestellt.

Was ist denn nun kaputt? Schauen Sie sich Zeile 7Ø an! 7Ø T = SQR (2H/G) – das entspricht doch genau der schlauen Formel aus dem Tabellenbuch?

Auch das Übertragen von Formeln hat offenbar seine Tücken. Computergerecht nämlich muß es sein! In der Formel $\sqrt{2H/G}$ lesen Sie ohne nachzudenken 2H als 2 MAL H !! Und genau so will es der Computer serviert haben!
Also verbessern Sie schleunigst:

```
7Ø T = SQR (2*H/G)
```

Jetzt sagt Ihnen der Computer prompt:

```
.451524          4.42945
```

Damit haben Sie zwar die Aufgabe gelöst – aber das Programm entspricht sicher noch nicht Ihren Ansprüchen! Es geht wieder an die Verzierungen!
Zunächst eine schicke Überschrift!

```
5 CLS
1Ø PRINT"BERECHNUNGEN NACH DEM GESETZ DES FREIEN FALLS"
2Ø PRINT" = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
= = = = "
3Ø PRINT
```

Nun verfeinern wir die Ausgabe:

```
9Ø PRINT"BEI FREIEM FALL AUS ";H;" METER"
95 PRINT
1ØØ PRINT"IST DIE FALLZEIT ";T;" SEKUNDEN"
11Ø PRINT"UND DIE ENDGESCHWINDIGKEIT ";V;" METER/SEKUNDE"
```

Bei einem Probelauf hat mir der Ausdruck nicht gefallen. Geben Sie noch einmal ein:

```
1ØØ PRINT"IST DIE FALLZEIT                    ";T;" SEKUNDEN"
```

Also nach ...ZEIT soviele Blanks, daß die Gänsefüßchen in 1ØØ und 11Ø schön untereinander stehen.
Und nun noch:

```
85 CLS
```

Wer kann sich schon etwas unter – beispielsweise – „40 m/s" vorstellen? An km/h sind wir gewöhnt. Also bieten Sie Ihren Freunden noch einen besonderen Service.

„40m/s" sind 40*3600 m/h – weil eine Stunde 3600 Sekunden hat. Das teilen Sie dann noch durch 1000 – weil 1 km 1000 m hat.
Für den Computer bedeutet das:

```
12Ø PRINT
13Ø PRINT"DAS IST EINE GESCHWINDIGKEIT VON ";V*3.6;" KM/STUNDE"
```

Und für den, der es öfter spielen will, fügen wir noch ein:

```
14Ø FOR X = 1 TO 1ØØØ : NEXT
15Ø GO TO 5
```

Was haben Sie dazugelernt?

- Achten Sie darauf, daß sie bei Produkten in Formeln und mathematischen Ausdrücken das * nicht vergessen!
  3H ; 5X/2B usw. ist für den Computer unverständlich. Er will es so haben: 3*H ; 5*X/2*B usw.
- Sagen Sie, daß Sie mir Zeile 13Ø nicht glauben?
  Das ist so gerechnet: V *(3600/1000) = V * 3.6. Einverstanden ?
- Eine Anmerkung noch für Perfektionisten: Für Höhen ab ca. 5000 m erreichen Sie so langsam Schallgeschwindigkeit...
  Die Formeln berücksichtigen nicht den Luftwiderstand, den der fallende Körper erfährt und der ihn verlangsamt. Die Formeln gelten (genau) nur für den Weltraum!

24 Zwischenexamen

...ist nicht der Titel für das nächste Programm. Nein, Sie sollen einmal versuchen, ein Programm selbständig zu schreiben.

Die folgenden Skizzen habe ich dem schon erwähnten Tabellenbuch entnommen (*Abb. 9*).

Ihre Aufgabe:

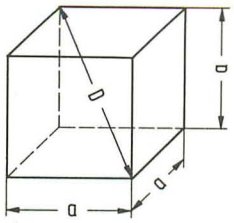
- Für die vier Körper Würfel, Prisma, Pyramide und Kegel sollen Sie ein Programm schreiben, welches das
- Volumen für beliebige Eingangsgrößen ermittelt.

Es wäre unfair, zu spicken; das vollständige Programm folgt gleich. Sie sollten es ruhig einmal abdecken und sich ans Werk machen. Zu Ihrer eigenen Kontrolle!

Ich will Ihnen einige Tips geben. Sie können das Programm mit Hilfe folgender Befehle aufstellen:

PRINT...
INPUT...
IF...THEN...
ON...GO TO
Der bekannten „Warteschleife" und schließlich
GO...TO...

```
5 CLS
10 PRINT"  ZWISCHENEXAMEN"
20 PRINT"  UEBUNGSPROGRAMM ZUR VOLUMENBERECHNUNG"
20 PRINT" ==========================================
====="

30 PRINT
40 PRINT"     AUSWAHLTABELLE"
50 PRINT
60 PRINT"WUERFEL ........................... BITTE 1 DRUECKEN !"
70 PRINT"PRISMA ............................ BITTE 2 DRUECKEN !"
80 PRINT"PYRAMIDE........................... BITTE 3 DRUECKEN !"
90 PRINT"KEGEL ............................ BITTE 4 DRUECKEN !"
100 INPUT N
110 IF N<1 THEN 150
120 IF N>4 THEN 150
130 ON N GO TO 200,300,400,500
```



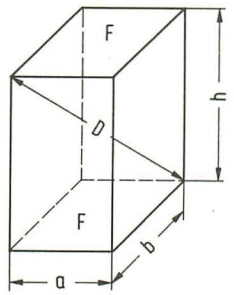
Würfel (Kubus)

Rauminhalt:

$V = a \cdot a \cdot a = a^3$

Kantenlänge: $a = \sqrt[3]{V}$

Raumdiagonale $D = a\sqrt{3}$



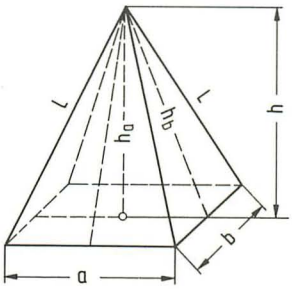
Prisma

V = Grundfläche × Höhe

$V = a \cdot b \cdot h \qquad V = F \cdot h$

$h = \frac{V}{F} \qquad a = \frac{V}{b \cdot h} \qquad b = \frac{V}{a \cdot h}$

$D = \sqrt{a^2 + b^2 + h^2}$



Pyramide

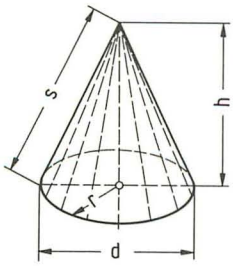
$V = \frac{a \cdot b \cdot h}{3} = \frac{F \cdot h}{3}$

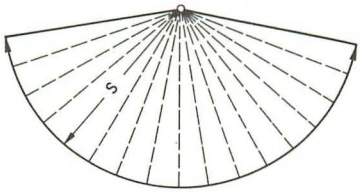
$h_b = \sqrt{h^2 + \frac{a^2}{4}}$

$L = \sqrt{h_b^2 + \frac{b^2}{4}}$

h_a und h_b = Flächenhöhen L = Kantenlänge

Kegel





$2\pi r = \pi \cdot d$

Mantelfläche $M = \frac{\pi \cdot d \cdot s}{2} \qquad V = \frac{F \cdot h}{3}$

Abb. 9 Lassen Sie den Computer bei der Berechnung der geometrischen Körper mitspielen

```
150 PRINT"FALSCHEINGABE !!"
160 FOR X = 1 TO 1000 : NEXT
170 GO TO 5
200 CLS
210 PRINT"WUERFELBERECHNUNG"
215 PRINT"= = = =
```

```
510 PRINT"KEGELBERECHNUNG"
520 PRINT"=========================================
======"
530 PRINT
540 INPUT"RADIUS DES GRUNDKREISES IN MM EINGEBEN ! ";R
550 INPUT"HOEHE DES KEGELS EINGEBEN ! ";H
560 F = R↑2*3.1416
570 PRINT"DAS VOLUMEN DER PYRAMIDE IST ";(F*H)/3;" KUBIKMILLI
    METER"
580 FOR X = 1 TO 1000 : NEXT
590 GO TO 5
600 END
```

So; war es schwer? Ich würde mich nicht wundern, wenn Sie ein Programm hingekriegt hätten, das bestens funktioniert – aber (ganz) anders aussieht, als meines! Machen Sie sich nichts draus. Merken Sie sich lieber:
10 Programmierer liefern für die gleiche Aufgabe mindestens 11 Programme!
Es kommt nicht (so sehr) darauf an, wie Sie Ihr Programm aufgebaut haben – es kommt (viel mehr) darauf an, daß es funktioniert!

Was haben Sie dazugelernt?

- Hoffentlich als erstes: Daß das Programmieren gar nicht so schwer ist.
- Daß dieses schon „lange" Programm – von den unterschiedlichen Formeln einmal abgesehen – im Grunde aus vier nahezu gleichen Programmteilen bestand.
  Diese wurden lediglich durch
  ON...GO TO...
  "WARTESCHLEIFE" und
  Rücksprung (GO...TO...)
  zusammengefügt.
- Hoffentlich haben Sie auch das Entflechten von Formeln richtig gemacht. Für alle Fälle noch mal eine kleine Wiederholung!
  Auf der Skizze mit den Formeln steht bei Pyramide (unter anderem)

$$L = \sqrt{h_b^2 + \frac{b^2}{4}}\,.$$

Schreiben Sie die doch mal (noch ist Zwischenexamen!) computergerecht. Hilfestellung: Auf Seite 29 haben Sie etwas über Klammern gehört. Machen Sie es wie der Computer, fangen Sie „innen" an.

h_b^2 wird zu HB↑2;

$\frac{b^2}{4}$ wird zu B↑2/4;

$h_b^2 + \frac{b^2}{4}$ wird zu (HB↑2)+(B↑2/4);

und nun noch
SQR ((HB↑2) + (B↑2 / 4))

Fertig. Oder?
Beachten Sie die Feinheiten der mathematischen Umgangssprache: hb würde heißen h mal b; für den Computer also h*b. Und da der keine Kleinschreibung kennt, hieße es H*B.

In der Formel steht aber h_b (das b ist tiefer gestellt!). Jetzt ist b ein Index von h, gesprochen „ha be", gewissermaßen „das b-te h" (Verzeihung!). Aber Sie machen keinen Fehler, wenn Sie dem Computer das als HB servieren!

25 Wird jetzt die Mathematik verbogen?

Das Symbol der „Gleichheit" wurde uns in der Schule am Beispiel der im Gleichgewicht befindlichen Waage erklärt, so, wie es die *Abb. 10* zeigt.

„A" und „5" sind „gleich", sind identisch, haben den gleichen Wert. Nicht mehr und nicht weniger soll das Gleichheitszeichen ausdrücken.

Nun werde ich Ihnen gleich im nächsten "Lernabschnitt" die Behauptung zumuten:

A = A + 1

Jedem Mathematiker sträubt sich beim Anblick dieser Aussage das Nackenhaar! Bevor sich Ihres auch sträubt, erinnern Sie sich – oder lesen nach auf Seite 48 – was ich Ihnen zum Thema Zuweisungen erklärt hatte:

Wenn es dort hieß:

...A = 7

Dann hatte das Gleichheitszeichen NICHT die Bedeutung von „gleich" oder „identisch" im Sinne der Mathematik.

Vielmehr bedeutet dieser Ausdruck bei der Programmierung:

... A = 7	A sei 7
... A = 7	lasse A gleich 7 sein!
oder	
... A = 7	weise A den Wert 7 zu!
usw.	

Das läuft zwar fast auf dasselbe hinaus. Es ist mir jedoch wichtig, daß Sie erkennen, wie hier – bei der Programmierung – „=" zu verstehen ist.

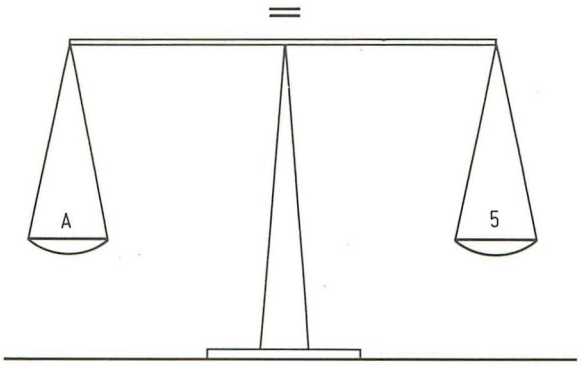


Abb. 10 Es gibt viele Arten der Gleichheit. Sie seien hier durch eine Waage symbolisiert

Nehmen Sie an, Sie finden ein Programm wie folgendes:

1Ø A = 1	hier wird also A der Wert 1 zugewiesen...
2Ø...	In den Zeilen 2Ø...4Ø hat der Computer irgendwas mit dem
3Ø...	Wert A = 1 zu tun – was uns jetzt nicht interessiert.
4Ø...	
5Ø A = A + 1	...trifft hier auf diese ungeheuerliche Behauptung...
6Ø...	

Prägen Sie sich bitte ganz fest ein, wie Zeile 5Ø gemeint ist, wie das zu lesen ist und wie das zu verstehen ist:

...nach dem Du – lieber Computer – mit der von mir in 1Ø vorgegebenen Größe A = 1 in den Zeilen 2Ø, 3Ø, 4Ø das Befohlene gemacht hast...

5Ø A = A + 1 ⟶ ERHÖHE NUN A UM 1!

Ab Zeile 5Ø hat also A für den Computer den Wert 2. Und er würde mit dem Wert 2 für A ab Zeile 6Ø im Programm fortfahren.

Ich empfehle Ihnen einen Rücksprung an den Anfang dieses Kapitels. Lesen Sie es noch einmal! Denn diese Programmierweise ist außerordentlich nützlich und vielseitig. Um das richtig einzusetzen, sollten Sie den Sinn des Gesagten ganz verstanden haben!

Wie gewohnt, wollen wir das durch einige Programme solange einüben, bis Ihnen das in Fleisch und Blut übergegangen ist:

```
NEW
1Ø A = 1
2Ø PRINT A , A*2 , A*3 , A*4
```

In Zeile 1Ø haben Sie A den Wert 1 zugewiesen...
In Zeile 2Ø haben Sie befohlen, was damit zu tun ist.

Jetzt:

3Ø A = A + 1	...hier soll A um 1 erhöht werden...
4Ø GO TO 2Ø	Hier befehlen Sie, mit dem neuen Wert für A (jetzt 2!) zu Zeile 2Ø zurückzugehen, um in 2Ø noch einmal die Rechnung auszuführen!

Das tut der Computer treu und brav; findet dann in Zeile 3Ø wieder:
Erhöhe A um 1.
Da für ihn A inzwischen 2 war, wird A damit zu 3.
Starten Sie einmal das Programm!
Ja, ja – ich weiß! Gleich glüht der Apparat! Drücken Sie BREAK! Sie haben gesehen, wie das funktioniert? Daß Sie dem noch keinen rechten Sinn abgewinnen können, verstehe ich gut.

Ergänzen Sie Ihr Programm um:

```
35 IF A = 15 THEN 5Ø
5Ø PRINT "ENDE"
```

Jetzt hat es wohl mehr Sinn?
Betrachten Sie *Abb. 11*, damit Sie das Grundmuster dieser wichtigen Programmvariation verstehen! *Abb. 12* zeigt die allgemeine Struktur der Schleife mit Filter (= Abfrage) am Ende.

Und nun ein aus dem Leben gegriffenes Beispiel, das ich mit einer kleinen Geschichte einleiten möchte.

Neulich hörte ich im Autoradio, daß sich in der letzten Tarifrunde der Arbeitgeberverband A mit der Gewerkschaft G auf folgenden neuen Tarifvertrag geeinigt haben (sinngemäß!):
– Die Löhne der oberen Lohngruppen (Facharbeiter) werden um 5 % angehoben.
– Die Löhne der unteren Lohngruppen (Hilfskräfte) werden um 6.5 % angehoben.
Ich erinnere mich daran, daß die Gewerkschaft G dieses Spiel schon seit Jahren treibt. „Das muß doch zur Nivellierung führen! Irgendwann gibt es keinen Unterschied mehr zwischen Fach- und Hilfsarbeiterlohn", dachte ich.

Und dann hat es gefunkt – und deshalb erzähle ich Ihnen diese Geschichte. Ich beschloß, in dieser SACHFRAGE endlich vom „DENKEN" und „MEINEN" und „GLAUBEN" zum „WISSEN" zu kommen – wozu habe ich schließlich so einen schlauen Computer?

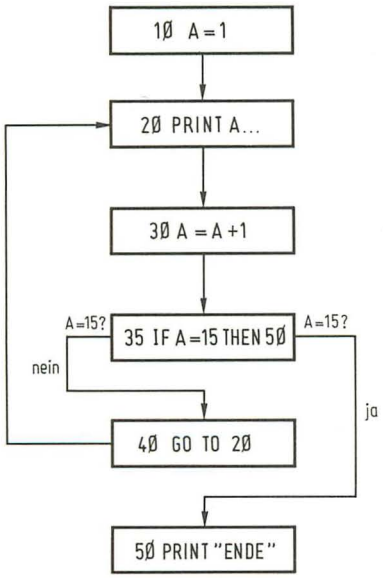


Abb. 11 Hier das Grundmuster, dem eine Schleife mit „Abfrage am Ende" folgt

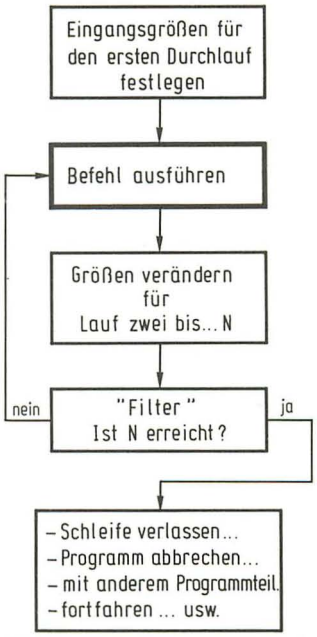


Abb. 12 Allgemeine Darstellung des Ablaufs von Schleifenoperationen

Zu Hause angekommen, setzte ich mich an meinen Computer, um den zu konsultieren. Der wußte natürlich – wie zu erwarten war – gar nichts! Also ging's darum, ein kleines Programm zu schreiben.

1. Schritt: Die Eingangsgrößen sind festzulegen. Ich habe mich für folgende entschieden: Angenommen, das finstere Spiel beginnt in diesem Jahr, gewissermaßen „im Jahr Ø". Das führt zu:

```
10 N = Ø
```

Der Facharbeiter hat heute einen bestimmten Stundenlohn (den ich nicht kenne). Ich wählte für den Facharbeiterlohn 100 (meinetwegen übersetzen Sie das mit 100 % seines heutigen Lohnes).

```
2Ø F = 1ØØ
```

Der Hilfsarbeiter verdient weniger; ich habe einmal angenommen, daß er heute 80 % des Facharbeiterlohnes erhält

```
3Ø H = 8Ø
```

und das wollte ich mir ausdrucken lassen.

```
4Ø PRINT F , H
```

2. Schritt: Jetzt müssen für das nächste und die folgenden Jahre die Größen N, F und H verändert werden.

```
5Ø N = N + 1
```

wie ich aus dem Radio erfahren hatte, erhält der Facharbeiter 5 % mehr. Das könnte man furchtbar kompliziert so ausdrücken:

1ØØ + (1ØØ *5) / 1ØØ ergibt
1ØØ + " 5 von 1ØØ".

Einfacher geht es so:

1ØØ * 1.Ø5

(Merken Sie sich diese „Kurzschrift" für die Prozentrechnerei! Das benötigen Sie immer wieder.)

```
6Ø F = F * 1.Ø5
```

sinngemäß heißt es dann beim Hilfsarbeiter – der ja 6.5 % mehr bekommt:

```
7Ø H = H * 1.Ø65
```

So weit – so gut. Was wollte ich eigentlich wissen?

Ich wollte wissen, nach wievielen Jahren der Hilfsarbeiter mehr verdient als der Facharbeiter – wenn dieses Erhöhungsspiel Jahr für Jahr mit den gleichen Prozenten gespielt wird. Also habe ich folgendes „Filter" in mein Programm eingebaut:

```
8Ø IF H>F THEN 1ØØ
```

Das heißt, der Computer soll solange rechnen, bis H größer ist als F. Oder anders herum: solange das nicht der Fall ist, soll er weiterrechnen. Das führt zu:

```
9Ø GO TO 4Ø
```

Für die Ausgabe habe ich mir ausgedacht:

1ØØ PRINT"NACH";N;"JAHREN VERDIENT DER HILFSARBEITER MEHR ALS DER FACHARBEITER!"

Ich darf annehmen, daß Sie inzwischen dieses Programm im Kasten haben? Wenn Sie es jetzt starten, wird ihnen der Computer am Ende zweier Zahlenkolonnen ausgeben:

NACH 16 JAHREN VERDIENT DER HILFSARBEITER MEHR ALS DER FACHARBEITER!

So, nun weiß ich's endlich!

Was haben Sie dazugelernt?

- Eine sinnvolle Erweiterung von Zuweisungen.
- Dieses Rechnen in Schleifen erinnert zwar an die FOR...NEXT...-Schleife. Aber bei dieser verändern Sie nur eine Variable. Ich habe hier drei Variablen verändern lassen. Nämlich N, F und H. Wenn die Aufgabe es erfordert, können Sie auch 30 nehmen.
- Sie können innerhalb der Schleifen die einzelnen Variablen unterschiedlich verändern. Ich habe im Sinne der Aufgabe gewählt.
  N = N + 1 also in jedem Lauf um 1 erhöht.
  F = F * 1.Ø5 also in jedem Lauf mit 1.Ø5 multipliziert.
  H = H * 1.Ø65 also in jedem Lauf mit 1.Ø65 multipliziert.
  Je nach Aufgabe könnten Sie auch bestimmen:
  N = N – 2 ; N = N / 3 usw.
- ZWEI TÜCKEN hat diese „Schleiferei"!
  – Sie müssen sorgsam beachten, wo und wie Sie *einsteigen*!
  – Sie müssen sorgsam beachten, wo und wie Sie *aussteigen*!
  Betrachten Sie die letzte Zahlenreihe Ihres Computerausdrucks: F = 207.9... ; H = 205.76...
  Da ist ja H NOCH NICHT GRÖSSER ALS F!!
  Zur Erklärung schauen Sie noch einmal auf das Schema, Seite 89. Der Computer hat beim 16. Durchlauf (N = 16) erkannt:
  F = 218.295 ; H = 219.134
  Dort ist jetzt H größer als F!
  Der Computer ist aber bei N = 16 ausgestiegen zu Zeile 1ØØ – wie befohlen – und hat die Werte für F , H nicht mehr hingeschrieben. Weil er nicht mehr zu Zeile 4Ø zurückgekehrt ist!

- Und was das Einsteigen angeht: Hätte ich mit N = 1 begonnen, hätte der Computer ausgegeben ...NACH 17 JAHREN !... und mich glatt angelogen! Es erfordert also einiges Nachdenken, wie Sie Anfangswerte und den Zeitpunkt des Programm-Abbruchs setzen!

Hätten Sie Lust, an diesem Beispiel noch etwas zu üben? Entwerfen wir zunächst ein Szenario!

Wir beginnen das Erhöhungsspiel im Jahre 1980. Natürlich merken die Facharbeiter langsam, daß sie – relativ – immer weniger verdienen. Sie reagieren entsprechend. Das führt dazu, daß beide Lohngruppen sich für die Jahre 1985 bis 1988 einschließlich mit je 5 % begnügen müssen. Dann erkennt man langsam, daß offenbar immer weniger Leute Lust haben, einen Beruf zu erlernen. Um dem sich abzeichnenden Facharbeitermangel abzuhelfen, dreht man den Spieß um und erhöht erstmals für das Jahr 1989 die Löhne der Facharbeiter um 7 %, die der Hilfsarbeiter um 5 %.
Preisfrage: Wie sehen H und F im Jahre 1995 aus?
Trauen Sie sich zu, diese Aufgabe zu lösen?

Fangen wir gemeinsam an:

```
10 N = 1980
20 F = 100
30 H = 80
40 PRINT N , F , H
50 N = N + 1
60 F = F * 1.05
70 H = H * 1.065
80 IF N = 1985 THEN 100
90 GO TO 40
```

So, bis hierher ist das Programm fast identisch mit dem vorigen. Jetzt müssen wir ab 1985 die Variablen F und H mit anderen Faktoren arbeiten lassen!

```
100 N = N
110 F = F
120 H = H
130 PRINT N , F , H
```

Hier müssen wir zwei Dinge tun:

1. Wir müssen die Größen aus dem ersten Lauf (bis ...90) unverändert in den zweiten Lauf (ab 100...) hinüberbringen. Denn es war zwar im ersten Lauf 1985 erkannt worden. Aber es wurde aus schon bekannten Gründen NICHT hingeschrieben. Durch N = N geschieht das jetzt erst im zweiten Lauf (...in 130).
2. Wir müssen in der zweiten Schleife auch (in 130) einen neuen PRINT-Befehl einbauen, da wir den in 40 nicht mehr benutzen können!

```
140 N = N + 1
150 F = F * 1.05
160 H = H * 1.05
170 IF N = 1988 THEN 190
180 GO TO 130
190 N = N
200 F = F
210 H = H
220 PRINT N , F , H
230 N = N + 1
240 F = F * 1.07
250 H = H * 1.05
260 IF N = 1996 THEN 280
270 GO TO 220
280 END
```

Erkennen Sie die Übereinstimmungen – aber auch die Unterschiede – in den einzelnen Programmteilen? Achten Sie auch auf Zeile 260! Mit diesem Trick erreichen wir, daß die Werte für das Jahr 1995 noch mit ausgedruckt werden.

BITTE LASSEN SIE DIESES PROGRAMM NOCH STEHEN!
(Wir arbeiten daran in etwa 15 Minuten weiter!)

Abb. 13 erläutert noch einmal die einzelnen Programmschritte.

Wenn man das, was in dem letzten Programm geschehen ist, *graphisch* darstellen möchte, dann kommt etwas dabei heraus, was *Abb. 14* zeigt.

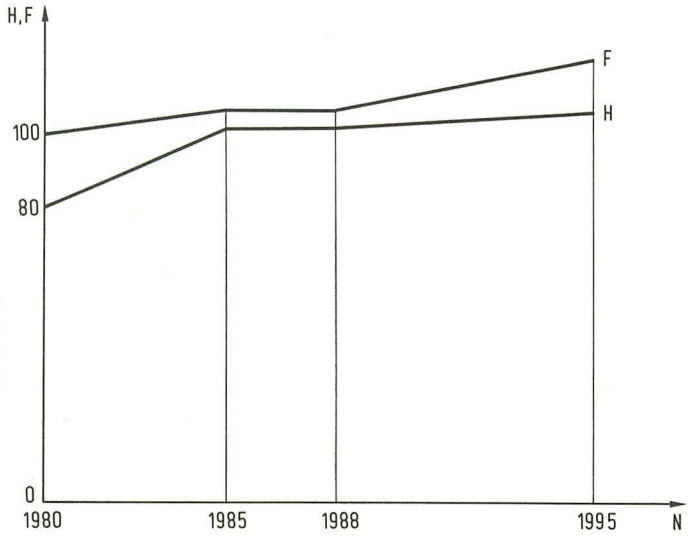


Abb. 14 Das Lohnniveau von Facharbeitern und Ungelernten als Grafik dargestellt. Die zugrundeliegenden Daten sind frei erfunden

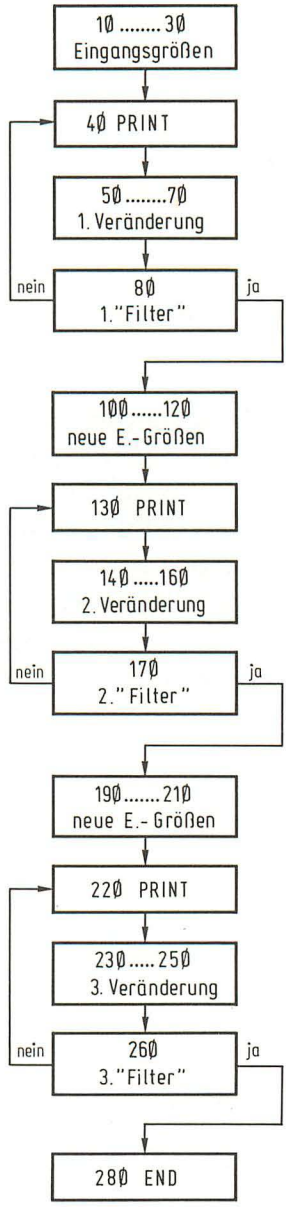


Abb. 13 Das Flußdiagramm zum Lohnniveau-Programm zeigt deutlich, daß die Schleifen mit Eingangswerten versorgt werden müssen und daß eine Kontrolle für den Abbruch vorhanden sein muß

26 Jetzt geht es rund!

Manchmal können die vorzüglichen Eigenschaften des Computers auch ausgesprochen lästig sein. Etwa dann, wenn er etwas – wie im vorigen Programm – auf sechs Stellen hinter dem Komma ausrechnet, was Sie nur auf zwei Stellen zu wissen wünschen.

Wenn Sie die Zahl 127.32938 sehen, aber nur an zwei Stellen hinter dem Komma interessiert sind, lesen Sie blitzschnell nur 127.33.

Dabei haben Sie die zweite Stelle hinter dem Komma bereits aufgerundet, weil Sie auf den ersten Blick erkannt haben, daß die dritte Stelle hinter dem Komma 9 ist. Sie haben also ...3**29** zu ...3**3** gemacht und liegen damit dichter bei dem wahren Wert ...329, als hätten Sie sich bloß für ...3**2** entschieden. So etwas sollte ein Computer halt auch können!

Er kann es. Allerdings nicht so problemlos wie Sie. Man muß ihm das mit einem trickreichen Programm erst beibringen. An dieses Programm wollen wir uns jetzt in kleinen Schritten heranmachen. Wir führen einen neuen Befehl namens INT ein. INT ist die Abkürzung vom „INTeger" – und das heißt zu deutsch „ganze Zahl".

Finden Sie heraus, wie das wirkt. Geben Sie ein: (ACHTUNG! Denken Sie an das Programm von vorhin! KEINE ZEILEN-NUMMER!)

PRINT INT (3.5) statt PRINT geht auch?

Und jetzt nicht RUN, sondern bloß [ENTER]

Der Computer gibt aus: 3.

Und nun:

PRINT INT (–3.5)

Der Computer gibt aus: –4.

Wenn Sie diesen vermeintlichen Unsinn verstehen wollen, blicken Sie auf die Zahlengerade der *Abb. 14a*.

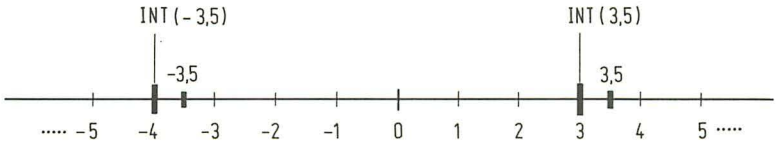


Abb. 14a Auf der Zahlengeraden sind einige Werte für INT (XX) eingetragen

In Worten bedeutet das:
Auf den Befehl INT(...) liefert der Computer...
...die *nächstliegende...*
kleinere...
...ganze Zahl der in ( ) hinter INT befindlichen Dezimalzahl.

Das müssen Sie mehrmals lesen und mit den beiden Beispielen vergleichen:

- Bei INT (3.5) ist in ( ) die Dezimalzahl 3.5 eingetragen.
  Der Computer macht daraus 3.
  3 aber ist die 3.5 *nächstliegende,*
  kleinere,
  ganze Zahl.
- Bei INT (–3.5) ist in ( ) die Dezimalzahl –3.5 eingetragen.
  Der Computer macht daraus –4.
  –4 aber ist die –3.5 *nächstliegende,*
  kleinere,
  ganze Zahl.

Wozu das gut sein soll, lernen Sie am besten an einem Programm, das wir zur Schonung des noch vorhandenen mit der Zeilen-Nummer 5ØØ beginnen lassen. Also:

```
5ØØ INPUT" ICH BIETE 7.Ø3 % ZINSEN.WIEVIEL KAPITAL WOLLEN SIE AN-
    LEGEN"
51Ø INPUT A
52Ø K = A * 1.Ø7Ø3
53Ø PRINT"NACH EINEM JAHR BESITZEN SIE ";K;" DM"
54Ø END
```

Wenn Sie jetzt einfach RUN eingeben, startet Ihr *gespeichertes* Programm, das ja mit einer niedrigeren Zeilen-Nummer beginnt. Wir wollen es überspringen und geben deshalb ein:

RUN 5ØØ

[ENTER]

Jetzt beginnt der Computer seine Arbeit erst ab Zeile 5ØØ!

Geben Sie bitte auf die Fragen des Computers nacheinander die unten stehenden Antworten ein. Sie erhalten dann:

Eingabe für A	Ausgabe für K
1ØØ	1Ø7.Ø3
97.37	1Ø4.215
9Ø	96.327

Sie haben bei der zweiten und dritten Eingabe ein Ergebnis mit drei Stellen hinter dem Komma, obwohl – wie bei Geldgeschäften üblich – Sie es nur „auf den Pfennig genau" haben möchten.

Ergänzen Sie Ihr Studienprogramm um:

525 K = INT (K)

Geben Sie die gleichen Zahlen wie vorhin ein; Sie erhalten:

Eingabe für A	Ausgabe für K
1ØØ	1Ø7
97.37	1Ø4
9Ø	96

Protest! Mit einer Bank, die so großzügig rundet, wollen Sie sicher nichts zu tun haben! Immerhin hat Sie Ihnen – im letzten Beispiel – ganze 32.7 Pfennig weggerundet – oder in die eigene Tasche gesteckt!

Vielleicht hilft uns das weiter?

525 K = INT (K* 1ØØ)/1ØØ

Immer noch mit den Zahlen von vorhin, ergibt das jetzt:

Eingabe für A	Ausgabe für K
1ØØ	1Ø7.Ø3
97.37	1Ø4.21
9Ø	96.32

Das sieht doch schon freundlicher aus!

Vollziehen wir den Computerbefehl nach (mit der Eingabe 97.37).

...in Zeile 52Ø wird K zu 1Ø4.215

...in Zeile 525 wird zunächst gerechnet K * 1ØØ

das ergibt 1Ø421.5

...dann INT (1Ø421.5) – das ergibt 1Ø421

...das geteilt durch 1ØØ – das ergibt 1Ø4.21

Geschafft? Nicht ganz! Denn bei dem Beispiel A = 9Ø ist herausgekommen K = 96.32.

Es wurde also abgerundet 96.327 zu 96.32 – und das entspricht nicht den Regeln eines honorigen Bankiers! 96.**33** würde der Ihnen nämlich gutschreiben.

Aber das kriegen wir auch noch hin! Schreiben Sie:

525 K = INT (K * 1ØØ + .5)/1ØØ

Und nun erhalten Sie für A = 9Ø tatsächlich K = 96.33!

Auch hier wollen wir den Rechengang nachvollziehen!

K = INT (K * 1ØØ + .5)/1ØØ:

erstens:	K * 1ØØ	=	9632.7
zweitens:	+ .5	=	9633.2
drittens:	INT(K..)	=	9633
viertens:	/1ØØ	=	96.33

Das geheimnisvolle Hinzufügen von .5 bewirkt also hier die gewünschte Aufrundung!

Zum Vergleich:

K	=	1Ø4.213
K * 1ØØ	=	1Ø421.3
+ .5	=	1Ø421.8

Wir kommen also hier durch Addition von .5 nicht von der Zehntel- in die Einerstelle!

.../1ØØ	=	1Ø4.21

Wenn Sie das Besprochene noch einmal auf einen Blick sehen wollen, geben Sie ein:

```
6ØØ FOR A = 1ØØ.ØØ1 TO 1Ø Ø.4 STEP .Ø33
61Ø PRINT A,INT(A),INT(A*1ØØ)/1ØØ ,INT(A*1ØØ+.5)/1ØØ
62Ø NEXT A
```

Auf RUN 6ØØ sehen Sie, daß nur in der vierten Spalte korrekt gerundet wird.

Und nun arbeiten wir – wie versprochen – am letzten Programm weiter und ergänzen es um:

```
 65 F  = INT(F*1ØØ +.5)/1ØØ
 75 H  = INT(H*1ØØ +.5)/1ØØ
155 F  = INT(F*1ØØ +.5)/1ØØ
165 H  = INT(H*1ØØ +.5)/1ØØ
245 F  = INT(F*1ØØ +.5)/1ØØ
255 H  = INT(H*1ØØ +.5)/1ØØ
```

Das ist doch jetzt ein professionelles Ergebnis?!

Merken Sie sich für den Umgang mit Geld die „klassische" Formel:

```
K = INT(K*1ØØ +.5)/1ØØ
```

Insbesondere in der Mathematik wollen Sie gelegentlich auch runden, aber nicht auf zwei Stellen, sondern auf drei oder vier oder mehr. Wie macht man nun so etwas? Sehen Sie noch einmal hin:

Das ergibt Rundung auf *zwei* Stellen.

```
K = INT ( K * 1ØØ + .5 ) / 1ØØ
              ↓          ↓
             1ØØ        1ØØ ist 10² (Zehn hoch zwei)
```

Das ergibt Rundung auf *drei* Stellen.

```
K = INT ( K * 1ØØØ + .5 ) / 1ØØØ
              ↓          ↓
             1ØØØ       1ØØØ ist 10³ (hoch drei)
```

Ganz allgemein gilt:

K = INT(K * 1Ø↑N + .5)/ 1Ø↑N ergibt Rundung auf N Stellen!

Hier biete ich Ihnen ein Programm zum Ausprobieren:

```
NEW
5 CLS
1Ø INPUT"WELCHE ZAHL WOLLEN SIE RUNDEN "; K
2Ø INPUT"AUF WIEVIELE STELLEN "; N
3Ø A = INT(K * 1Ø↑N + .5) / 10↑N
35 PRINT
4Ø PRINT K;" AUF";N; " STELLEN GERUNDET, ERGIBT "; A
5Ø FOR X = 1 TO 1ØØØ : NEXT
6Ø GO TO 5
```

Wenn Sie genug gespielt haben, noch etwas Ernsthaftes zu INT. Es kommt vor, daß Sie für eine bestimmte Aufgabe aus einer Dezimalzahl (Beispiel: 6.5) die EINER als 6, die Zehntel als 5 isolieren sollen. Ich zeige Ihnen, wie man das mit INT macht.

Programm	Was geschieht?
NEW	
1Ø INPUT N	Sie geben 6.5 ein!
2Ø E = INT(N)	E (für EINER) ist dann 6
3Ø Z = (N * 1Ø)–(E * 1Ø)	Z (für ZEHNTEL) ist: N * 1Ø = 65; E * 1Ø = 6Ø; also: Z = 5!
4Ø PRINT E , Z	Ergebnis: 6 5.

Etwas verzwickter wird es, wenn Sie aus der Zahl 12.3 Zehner, Einer und Zehntel nach dem gleichen Schema isolieren wollen. Fangen Sie an:

Programm	Was geschieht?
1Ø INPUT N	Sie geben ein: 12.3
Zur Ermittlung der ZEHNER führen Sie eine Zwischenrechnung durch:	
2Ø A = INT(N)/1Ø	das ergibt 1.2
3Ø ZE = INT(A)	ZE für ZEHNER ergibt dann 1
Für die EINER programmieren Sie:	
4Ø E = (A * 1Ø) – (ZE * 1Ø)	das ergibt: (A*1Ø) = 12 (ZE*1Ø) = 1Ø Also E für EINER = 2
Und schließlich für die ZEHNTEL:	
5Ø Z = N*1Ø – INT(N)*1Ø	N*1Ø = 123 INT(N) = 12 ..*1Ø = 12Ø also Z für ZEHNTEL = 3

Und nun alles zusammen:
6Ø PRINT N , ZE , E , Z
ergibt:

```
12.3          1          2          3
```

Wir werden von dieser pfiffigen Technik noch Gebrauch machen! Sie können ja mal versuchen, ein Programm für das Isolieren der in 123.4 steckenden Ziffern zu schreiben!

Was haben Sie dazugelernt?

- Wie man den INT-Befehl zum Runden auf beliebig viele Stellen nutzen kann.

  Ebenso wie die „Warteschleife" sollten Sie – wenn's ums Geld geht – im Kopf haben:

  X = INT(X*1ØØ +.5)/1ØØ
- Gerade beim Üben hat man oft mehrere Programme in seinem Kasten. Um diese Programme oder Programmteile nicht gegenseitig zu stören, legen Sie in den Programm-Nummern der einzelnen Teile einen deutlichen Abstand ein. Also:

  Teil 1 5..... 27Ø END
  Teil 2 5ØØ..... 58Ø END
  Teil 3 7ØØ.....1ØØØ END
  usw.

  Abarbeiten lassen Sie diese Teilprogramme durch den Aufruf:

  RUN
  RUN 5ØØ
  RUN 7ØØ
  usw.
- Schließlich kann man den INT-Befehl nebst einigen Tricks dazu benutzen, aus Dezimalzahlen die einzelnen Stellen als Einzelziffern zu isolieren. Dazu müssen Sie ggf. Zwischengrößen für Zwischenrechnungen einführen. In unserem Beispiel war das:

  2Ø A = INT(N)/1Ø

27 Der Computer als Digitaluhr

Dieses Programm habe ich zu Ihrer Entspannung gedacht. Oder zum Trost: Falls Sie von der Programmiererei genug haben, brauchen Sie Ihren Computer nicht wegzuwerfen. Sie können ihn als recht genaue Digitaluhr weiterverwenden. Den Umbau werden wir ohne Lötkolben und Schraubenzieher durchführen – er erfolgt nur per SOFTWARE, wie die Profis das nennen.
Erster Schritt:

```
NEW
10 A = 1
20 PRINT A
30 A = A + 1
40 GO TO 20
```

Jaja, ich weiß, daß Sie sich durch dieses Programm fast veralbert fühlen. Aber es bildet die „wissenschaftliche Grundlage" unserer Digitaluhr und sie sollten es getrost einmal starten.

Wenn Sie genug haben, unterbrechen Sie mit [BREAK] und fahren Sie fort:

```
15 CLS
40 GO TO 15
```

So, ich hoffe, jetzt lachen Sie nicht mehr! Sie lernen, was man mit einem „gezielt" eingesetzten CLS NOCH machen kann, statt nur die Tafel auszuwischen!
Und jetzt noch:

```
35 FOR X = 1 TO 1000 : NEXT
```

Ich könnte mir vorstellen, daß Sie jetzt schon ahnen, wie es weitergeht. Das, was sich auf dem Bildschirm abspielt, sieht aus wie ein laufender Zähler. Und was ist eine Uhr anders als ein (Zeit)-Zähler?

Was wir als nächstes tun müssen, ist offensichtlich: Wir müssen einen „Zähltakt" gewinnen, der uns genau im Abstand einer Sekunde unseren Zähler weiterschaltet. Im Augenblick wird der Zähler – wie Sie ja selbst erarbeitet haben – durch die (zeitliche) Länge der FOR...NEXT-Schleife in Zeile 35 gesteuert. Um genau zu sein: wir müssen 1000 solange verändern, bis das „Weiterschalten" genau alle Sekunde passiert. Dafür gibt es keine feste Regel, weil das fast bei jedem Computer anders ist. Bei meinem hatte ich für den Anfang einen ziemlich genauen Sekundentakt durch:

```
35 FOR X = 1 TO 500 : NEXT
```

Schon jetzt könnten Sie Ihr Programm in Richtung Stoppuhr weiterentwickeln, mit dem Sie nur Sekunden – oder, bei entsprechend „kurzer" FOR...NEXT-Schleife – Zehntelsekunden zählen können. Aber wir haben uns ein ehrgeizigeres Ziel gesetzt: wir wollen ein Gerät bauen, das neben Sekunden auch Minuten und Stunden zählt.

Zweiter Schritt:

```
NEW
10 S = 59
20 INPUT"GEBEN SIE DIE AKTUELLE MINUTE EIN ";M
30 INPUT"GEBEN SIE DIE AKTUELLE STUNDE EIN ";ST
40 FOR N = 1 TO 500 : NEXT
50 S = S+1
60 IF S = 60 THEN 80
70 GO TO 500
80 S = 0
90 M = M + 1
100 IF M = 60 THEN 120
110 GO TO 500
120 M = 0
130 ST = ST + 1
140 IF ST = 24 THEN 180
150 GO TO 500
180 S = 0 : M = 0 : ST = 0
500 CLS
510 PRINT ST;":";M;":";S
520 GO TO 40
```

Die Wirkungsweise dieses Programms will ich Ihnen genau erklären. In Zeile 10 habe ich mit S = 59 zunächst die Sekunden mit 59 vorgegeben.

Die Zeilen 20 und 30 dienen dem Stellen der Uhr. Wenn Sie das Programm starten, wird der Computer beginnen:

GEBEN SIE DIE AKTUELLE MINUTE EIN

Angenommen, jetzt, wo Sie vor Ihrem Computer sitzen und die Uhr starten, ist es 11:10:15 – das heißt, 11 Uhr zehn und 15 Sekunden. Sie beantworten also die Frage des Computers mit 10. Auf [ENTER] wird er weiterfragen:

GEBEN SIE DIE AKTUELLE STUNDE EIN

Sie antworten mit 11, drücken aber *nicht* [ENTER]. Inzwischen wird es wohl 11:10:50 geworden sein. Sie beobachten Ihre Vergleichsuhr, und in dem Augen-

blick, wo der Sekundenzeiger von 58 auf 59 geht, drücken Sie [ENTER]. Damit fährt das Programm ab.

In Zeile 4Ø wird zunächst eine Sekunde gewartet. In Zeile 5Ø wird S um 1, also auf 6Ø, erhöht. Damit ist Zeile 6Ø „wahr" und der Computer springt zu Zeile 8Ø, wo S wieder auf Ø gesetzt wird.

In Zeile 9Ø angekommen, wird M um 1 erhöht, erreicht also 11. Zeile 1ØØ ist damit noch nicht „wahr" und Zeile 11Ø schickt den Computer zu 5ØØ, damit zunächst der Schirm gelöscht wird. Zeile 51Ø befiehlt nun, ST, M und S hinzuzuschreiben. Nach dem akuten Stand bedeutet das 11:11:0. Zeile 52Ø schließlich schickt das Programm wieder an den Anfang.
S wird um 1 auf 1 erhöht... Zeile 6Ø ist jetzt *nicht* wahr und der Computer geht zu 5ØØ, löscht den Schirm und schreibt: 11:11:1, usw.

Ist es 11:59:59 geworden, dann wird beim nächsten Durchlauf 6Ø und 1ØØ als wahr erkannt und es geschieht:

S wird auf Ø gesetzt
M wird auf Ø gesetzt
ST wird (in 13Ø) um 1 erhöht
ausgeschrieben wird 12:Ø:Ø.
Ist es 23:59:59 geworden, dann würde mit der nächsten Sekunde die Uhr auf 24... springen. „24 Uhr" aber gibt es nicht! Sobald also 24 erkannt wird, geht der Computer in Zeile 18Ø und setzt ALLES auf Ø. Ausgedruckt wird dann Ø:Ø:Ø. Und das Spiel beginnt von neuem.

Ist doch unterhaltsam, was man mit einem Computer alles anstellen kann! Wenn Sie sich von Ihrem Programm trennen können, machen wir mit dem nächsten Abschnitt weiter!

28 Der Computer liest Daten

Schreiben Sie einmal folgendes Programm:

```
NEW
10 READ A
20 PRINT A , A*2.5 , A*3.5 , A*4.5
30 GO TO 10
40 DATA 1,2,3
```

Dieses Programm druckt aus:

1	2.5	3.5	4.5
2	5	7	9
3	7.5	10.5	13.5

```
? OD ERROR IN 10
READY
>-
```

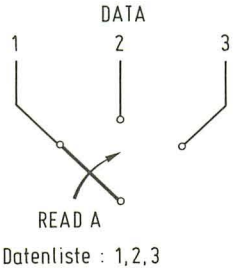


Abb. 15 Die READ-Anweisung schaltet intern im Computer einen Zeiger auf das nächste Datum weiter – bis nichts mehr da ist. Dann wird protestiert

In diesem neuen Programm gehören die Zeilen 10 und 40 zusammen.

READ.....DATA.....

heißt zu deutsch: LESE.....(für READ)
die DATEN.....(für DATA)

Das Programm oben befiehlt dem Computer:

10 READ...	Lese (oder hole oder suche) die Daten, die ich im Programm unter der Variablen A in der Datenliste abgelegt habe...
20 PRINT...	Führe den Print-Befehl aus. Der Computer geht in die DATA...-Zeile, findet dort für A als erstes den Wert 1 und führt damit den Print-Befehl aus.
30 GO TO 10	fordert ihn auf, das Spiel noch einmal zu spielen. Er findet in der Datenzeile als nächstes den Wert 2 (der Computer hat sich gemerkt, daß er 1 schon bearbeitet hat!).

Das tut der Computer so lange, wie er Daten findet.

Hat er alle Daten vorgekramt – und entsprechend Zeile 20 bearbeitet –, druckt er aus:

? OD ERROR IN 10

Das soll heißen: OUT OF DATA zu deutsch: „keine Daten mehr da!"

Die *Abb. 15* soll an einem Beispiel deutlich machen, wie das Zusammenspiel von READ und DATA vor sich geht. Abgebildet ist das Schaltbild eines elektri-

schen Schalters, hier das eines einpoligen, dreistufigen Drehschalters. Für jeden Durchlauf wird der Schalter um eine Position weitergedreht; dabei zeigt der Schaltfinger des Schalters jeweils auf das entsprechende Datum. Also:

Schaltstellung 1	entspricht erstem Lauf:	Datum ist 1.
Schaltstellung 2	entspricht zweitem Lauf:	Datum ist 2.
Schaltstellung 3	entspricht drittem Lauf:	Datum ist 3.

Das sollten Sie sich ruhig genau betrachten, damit Ihnen bereits an diesem ersten, einfachen Beispiel das Zusammenspiel von READ... und DATA... klar wird – es wird gleich komplizierter!

Schreiben Sie ein neues Programm!

```
NEW
10 READ A,B
20 PRINT A, A*5, B, B*5
30 GO TO 10
40 DATA 1,2,10,20,100,200
```

Dieses Programm liefert das Ergebnis:

1	5	2	10
10	50	20	100
100	500	200	1000

Das ist nicht auf Anhieb einzusehen. Wir wählen wieder unseren Schaltervergleich, verwenden aber diesmal einen zweipoligen Drehschalter mit drei Schaltstellungen (*Abb. 16*).

Obwohl die Datenliste lautete: 1,2,10,20,100,200, hat der Computer die Datenliste wie folgt interpretiert:

Für den ersten Lauf:	A = 1,	B = 2;
für den zweiten Lauf:	A = 10,	B = 20;
für den dritten Lauf:	A = 100,	B = 200.

In der Abbildung oben sind die Schleifer der beiden Schalterebenen starr miteinander verbunden, was durch die gestrichelte Linie angedeutet ist.

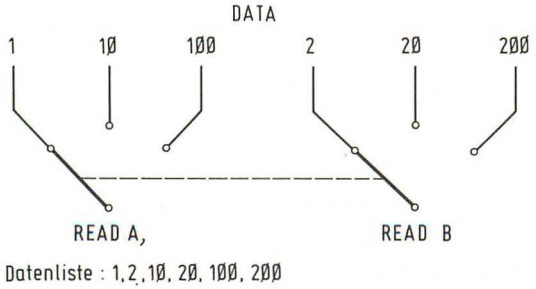


Abb. 16 So könnte ein Mechanismus aussehen, der wie eine READ A,B-Anweisung funktioniert, wenn sie mehrfach durchlaufen werden soll

READ A und READ B geschehen also gleichzeitig; an den einzelnen Schalterausgängen ist oben angeschrieben, was für Daten jeweils gelesen werden.

Weil es so schön war, geben Sie noch einmal ein:

```
NEW
10 READ A , B , C
20 PRINT A
30 PRINT A,B
40 PRINT A,B,C
50 GO TO 10
60 DATA 1,2,3,10,20,30,100,200,300
```

Dieses Programm führt den Computer zu folgendem Ergebnis:

```
1
1          2
1          2          3
10
10         20
10         20         30
100
100        200
100        200        300
? OD ERROR IN 10
```

Zum Verständnis dieses Computerergebnisses wählen wir wieder unseren Schaltervergleich. Sie vermuten richtig: dieses Mal wählen wir einen *drei*poligen Schalter mit *drei* Schaltstellungen (*Abb. 17*).

Beachten Sie, an welchen Schalterausgängen welche Daten angeschrieben sind und wenden Sie sich noch einmal dem Computerausdruck zu!

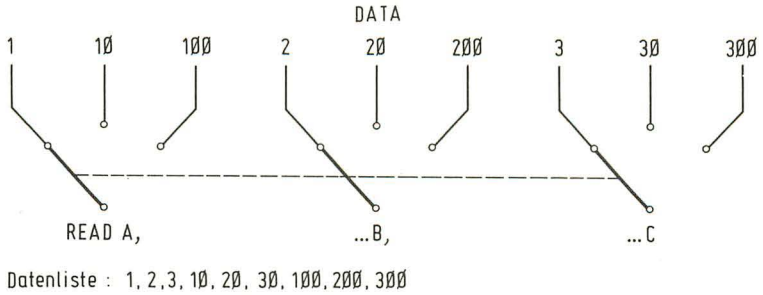


Abb. 17 Ein Mechanismus für READ A,B,C

Erster Durchlauf:
Der Schalter steht in der ersten Position. Der Befehl in Zeile 2Ø lautet: PRINT A. Der Schleifer für A zeigt auf 1 – also druckt der Computer 1.
Dann folgt Zeile 3Ø PRINT A,B. Der Computer ist noch beim ersten Durchlauf – d. h., die Schalterstellung ist noch unverändert. Also druckt der Computer
1 2.
Es folgt Zeile 4Ø PRINT A,B,C. Immer noch im ersten Durchlauf, druckt der Computer
1 2 3.
Erst jetzt folgt in Zeile 5Ø der Befehl, das Spiel von neuem zu beginnen.

Zweiter Durchlauf:
In unserem Schalterbeispiel bedeutet der Befehl in Zeile 5Ø GO TO 1Ø, daß der Schalter in die nächste Position gedreht wird. Folglich druckt der Computer auf den Befehl in Zeile 2Ø PRINT A nun den Wert 1Ø. Und so geht es weiter!

Dritter Durchlauf:
Sobald der Computer wieder Zeile 5Ø erreicht, bedeutet das für unser Bildbeispiel, eine Schalterstellung weiter zu schalten.
Schaltfinger A zeigt auf 1ØØ
Schaltfinger B zeigt auf 2ØØ
Schaltfinger C zeigt auf 3ØØ
Und das Spiel läuft von neuem ab!

Wenn Sie das durch ein paar einfache Übungen noch vertiefen möchten, dann löschen Sie in Ihrem Programm die Zeilen 3Ø und 4Ø (ich unterstelle, daß Sie inzwischen wissen, wie das geht!). Das Ergebnis wird sein:

1
1Ø
1ØØ

Schreiben Sie nun:

2Ø PRINT C

Der Computer wird ausgeben:

3
3Ø
3ØØ

Bleiben Sie zum Verständnis immer dicht an Ihrem Schalterbeispiel (*Abb. 17*).
Nun machen Sie etwas Neues:

```
1Ø READ A
2Ø PRINT A
```

Darauf wird der Computer antworten:

```
1
2
3
1Ø
2Ø
3Ø
1ØØ
2ØØ
3ØØ
```

Immer Ihre Schalterbeispiele fest im Blick, werden Sie wohl dahinterkommen, wie das Zusammenspiel von READ... und DATA... funktioniert!

- Mit READ... bestimmen Sie, welche Variablen-Namen die Daten in der Datenliste erhalten sollen.
- Bei READ A... sind ALLE Daten nacheinander A... das entspricht unserem *ein*poligen Schalter.
- Bei READ A,B haben Sie den *zwei*poligen Schalter kreiert. Jetzt wird definiert: erstes Datum = A, zweites Datum = B. Beim nächsten Durchlauf dann: drittes Datum = A, viertes Datum = B, usw.

Hoffentlich sind nun alle Klarheiten beseitigt?

Was haben Sie dazugelernt?

- Das Zusammenspiel der kombinierten Befehle READ... DATA...
- In meinen Beispielen hatte ich die Datenlisten immer an das ENDE des Programms gestellt. Wo Sie die Daten unterbringen – an den Anfang des Programms, irgendwo in die Mitte oder – wie ich – an das Ende – ist dem Computer gleichgültig.
  Immer wird er das ganze Programm nach DATA... absuchen, sobald er auf den Befehl READ... trifft. Ich will Sie nicht beeinflussen – aber mir gefallen die Daten am Ende eines Programms am besten; ich halte das für übersichtlicher!
- Wieviele Daten Sie in einem Programm unterbringen, hängt allein von der jeweiligen Aufgabe und davon ab, ob Ihr Computer genügend Speicherplatz hat. Ich habe mich für die Beispiele mit wenigen Daten begnügt. Ich hätte auch

zunächst in eine Zeile so viele Daten schreiben können, wie darauf (Bildschirm!) Platz haben. Reicht das nicht aus, könnte man eine neue Zeile vollschreiben. Und noch eine. Und noch eine...

- Wichtig ist, daß Sie immer zwischen zwei Daten ein Komma machen!
- Nach dem letzten Datum darf kein Komma stehen!
- Bei EINER Variablen (READ A) liest der Computer ALLE – beispielsweise 1ØØ – gespeicherten Daten nacheinander als A.
- Bei MEHREREN Variablen in der READ...-Zeile, zum Beispiel READ A,B,C,D, müssen Sie in der DATA-Liste mindestens VIER – und zwar je ein Datum für A, B, C und D – anbieten. Oder allgemein: Die Anzahl der Daten muß durch die Anzahl der Variablen in der READ...-Zeile teilbar sein.
  Wenn Sie beispielsweise die Variablen READ A,B,C,D festlegen, und nur 39 Daten zur Verfügung stellen, dann findet der Computer 9 mal Daten für alle vier Variablen. Beim 1Ø. Durchlauf findet er nur ein Datum für A, B und C – das 4Ø. fehlt. Er nimmt sich dann die Freiheit, eigenmächtig für das fehlende, 4Ø. Datum den Wert Ø zu setzen!

29 Der Computer liest immer noch Daten

Ganz zu Anfang hatte ich Ihnen beigebracht, daß nicht nur Zahlen Daten sind – auch Strings können vom Computer als Daten erkannt und verarbeitet werden. Machen wir die Probe auf das Exempel:

```
10 READ A$
20 PRINT A$
30 GO TO 10
40 DATA "SONNTAG","MONTAG","DIENSTAG","MITTWOCH",
   "DONNERSTAG"
50 DATA "FREITAG","SONNABEND"
```

Der Computer gibt aus:

```
SONNTAG
MONTAG
DIENSTAG
MITTWOCH
DONNERSTAG
FREITAG
SONNABEND
```

Zahlen und Strings können auch gemischt angeboten und verarbeitet werden – wenn Sie darauf achten, daß in der READ...-Zeile der Computer entsprechend vorbereitet wird.

Versuchen Sie es mal mit:

```
10 READ A$,A
20 PRINT A$,A
30 GO TO 10
40 DATA "SONNTAG =",1,"MONTAG =",2
50 DATA "DIENSTAG =",3,"MITTWOCH =",4
60 DATA "DONNERSTAG =",5,"FREITAG =",6
70 DATA "SONNABEND =",7
```

Der Computer wird Ihnen liefern:

```
SONNTAG = 1
MONTAG = 2
DIENSTAG = 3
MITTWOCH = 4
```

DONNERSTAG = 5
FREITAG = 6
SONNABEND = 7

Hier biete ich Ihnen nun ein Programm, das Ihnen nicht nur noch einmal READ...DATA... vorführt. Es zeigt Ihnen auch, daß man innerhalb von READ... DATA... mehr tun kann, als bloß rechnen – wie wir das etwa in den Beispielen mit PRINT A*2.5 – exerziert hatten.

```
10 PRINT "NAME",,,"ALTER"
20 PRINT" = = =
```

Da der Computer *schrittweise* durch das Programm geht – Sie sollten sich an den Stufenschalter erinnern! – wird er "ENDE" erst als letztes N$ finden. Sie haben das ja auch aus gutem Grund an das Ende gesetzt!

Zeile 4Ø: ...und nun soll der Computer in der Liste das A raussuchen!

Zeile 5Ø: Hier befehlen Sie: "sobald Du bei Deiner Suche ein A findest, das kleiner als 2Ø ist, dann schreibe mir dieses A und das dazu gehörende N$ hin!".

Zeilen 3Ø, 5Ø: Früher hatten wir oft geschrieben:
1ØØ IF...THEN 11Ø
11Ø PRINT...
Hier sind die beiden Zeilen zu einer zusammengezogen.
Zu: ...IF...THEN...PRINT...
Man lernt halt immer noch dazu!

Und nun beobachten wir den Computer bei der Arbeit! Er findet in der ersten Datenzeile "DANIEL",2
Er erkennt "DANIEL" ist NICHT "ENDE"...
und 2 IST kleiner als 2Ø...
Also sind die Voraussetzungen erfüllt, DANIEL 2
hinzuschreiben.
Beim nächsten Durchlauf findet er "EMIL",3Ø.
Seine Prüfung ergibt: "EMIL" ist NICHT "ENDE"..., aber
3Ø ist NICHT kleiner als 2Ø...
Also wird er befehlsgemäß darauf verzichten, etwas aufzuschreiben. Und so hangelt sich unser Apparat durch das Programm. Bis er findet: N$ IST GLEICH "ENDE" – und prompt schreibt er: ENDE DER LISTE.

Und weiter geht es mit dem Dazulernen! Bisher hatten wir mühevoll eine Datenliste aufgestellt, und diese nur einmal nach ganz bestimmten Befehlen (wie PRINT, oder PRINT...*2.5) durcharbeiten lassen. Nun lernen Sie, daß man EINE Datenliste auch MEHRMALS – und zwar immer auf andere Weise – durcharbeiten lassen kann. Dazu führen wir einen neuen Befehl ein: RESTORE.

Eine wörtliche Übersetzung bringt uns nicht viel; übersetzen Sie das sinngemäß mit: „Gehe noch einmal in die Datenliste zurück!" Sie lernen das an folgendem Programm:

```
NEW
5 PRINT"ERSTER LAUF DURCH DIE DATENLISTE"
8 PRINT" = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
= = = = "
1Ø READ A,B,C,D
2Ø PRINT A, A*1Ø
```

```
30 PRINT B,B*10
40 PRINT C,C*10
50 PRINT D,D*10
55 PRINT"ZWEITER LAUF DURCH DIE DATENLISTE"
58 PRINT" = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = "
60 RESTORE
70 PRINT A,A*100
80 PRINT B,B*100
90 PRINT C,C*100
100 PRINT D,D*100
110 PRINT"DRITTER LAUF DURCH DIE DATENLISTE"
120 PRINT" = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = "
130 RESTORE
140 PRINT A,A*1000
150 PRINT B,B*1000
160 PRINT C,C*1000
170 PRINT D,D*1000
180 DATA 1,2,3,4
```

Dieses Programm führt der Computer wie folgt aus:

```
ERSTER LAUF DURCH DIE DATENLISTE
=============================================
1          10
2          20
3          30
4          40
ZWEITER LAUF DURCH DIE DATENLISTE
=============================================
1          100
2          200
3          300
4          400
DRITTER LAUF DURCH DIE DATENLISTE
=============================================
1          1000
2          2000
3          3000
4          4000
```

Dieses Programm erklärt sich zwar (fast) von selbst – aber ich will Ihnen trotzdem noch einige Kommentare geben.

Zeilen 1Ø–58: Die sind so, wie Sie das bis jetzt hinreichend geübt haben.

Zeile 6Ø: Hier taucht zum ersten Mal RESTORE auf.
Wie Sie an dem Computerausdruck erkennen, geht der Computer noch einmal an den ANFANG der Datenliste. (Er war in Zeile 5Ø ja schon an deren ENDE!) RESTORE stellt also – um beim Schalterbeispiel zu bleiben – den Schaltfinger wieder in die Ausgangsposition.

Zeilen 7Ø–12Ø: ABER: Mit den ALTEN Daten arbeitet er jetzt die NEUEN Befehle ab!

Zeile 13Ø: Ein erneutes RESTORE scheucht ihn noch einmal – jetzt zum dritten Mal – in die Datenliste und veranlaßt die Abarbeitung der Befehle 14Ø–17Ø.

Was haben Sie dazugelernt?

- In dem READ...DATA...-Befehl akzeptiert der Computer auch Strings als Daten.
- Sie können Strings und Zahlen gemischt in einem Programm als Daten verwenden.
- Numerische Variable hatte ich in der DATA-Liste „nur so" hingeschrieben.
  ...DATA 1,2,3
  Das ist in Ordnung!
- Enthielt die DATA-Zeile Strings (Namen), dann hatte ich diese in " " gesetzt. Das war eine Vorsichtsmaßnahme; bei den meisten Computern können Sie darauf verzichten.
  ...DATA INGE,KLAUS,DANIEL
- Enthält der String Operationszeichen, dann müssen Sie ihn in jedem Falle in " " setzen!
  ...DATA "1.1.84","12.5 DM","BESTELL.-NR."
- Ordnen Sie die Daten in der DATA-Zeile so an, wie sie in der READ-Zeile gelesen werden sollen!
  ...READ A,A$
  ...DATA 1,DANIEL,2,KLAUS,3,ILSE

- RESTORE veranlaßt den Computer, nach Abarbeitung der Datenliste in diese zurückzukehren und mit den gespeicherten Daten nach neuen Befehlen wiederholt zu arbeiten.

30 Was ist ein Unterprogramm?

Ich glaube, Sie haben mal wieder eine kleine Entspannung verdient! Trotzdem werden Sie etwas dazulernen, nämlich: was ist ein Unterprogramm?

Wie der Name schon sagt: ein Programm, das sie UNTER ein anderes, unter ein HAUPT-Programm mischen! Am Beispiel lernen Sie das am schnellsten! Geben Sie ein:

```
10 PRINT"IN MUENCHEN STEHT EIN HOFBRAEUHAUS"
```

Und nun geht es ab ins Unterprogramm! Sie veranlassen das mit einem besonderen Befehl:

GOSUB..... das heißt: gehe ins UNTER...- ins SUB...-Programm!

Also:

```
20 GOSUB 200
```

GOSUB allein genügt nicht. Sie müssen dem Computer ein Ziel geben, Sie müssen ihm sagen, wo das Unterprogramm steht. Im Beispiel in Zeile 200! Dort möge stehen:

```
200 PRINT"EINS,ZWEI,GSUFFA!"
```

Sie haben also in Zeile 200 befohlen, der Computer soll das nur für Eingeweihte korrekt Sprechbare hinschreiben. Das tut der Computer auch, wie Sie gleich sehen werden. Aber dann müssen Sie ihm natürlich sagen, wie es weitergeht.

Sie sagen:

210 RETURN das heißt: Gehe zurück! In diesem Fall ins Hauptprogramm. An dem basteln wir wie folgt weiter:

```
30 PRINT"DA SCHAUT SO MANCHES MADEL RAUS!"
```

Und weil es so schön war:

```
40 GOSUB 200
```

Weiter kenne ich den Text nicht. Und ehe ich mich mit den Bayern anlege, weil ich mich an ihrem Liedgut vergreife, fahre ich fort:

```
50 PRINT
60 PRINT"WIE HEISST DER REFRAIN?"
70 PRINT
80 GOSUB 200
```

```
85 PRINT
90 PRINT"WEIL DAS SO SCHOEN IST : NOCH EINMAL !!"
95 PRINT
100 GOSUB 200
110 END
```

Bitte tippen Sie doch einmal LIST, und schauen Sie sich dieses Superprogramm an! Und nun lassen Sie es abfahren!

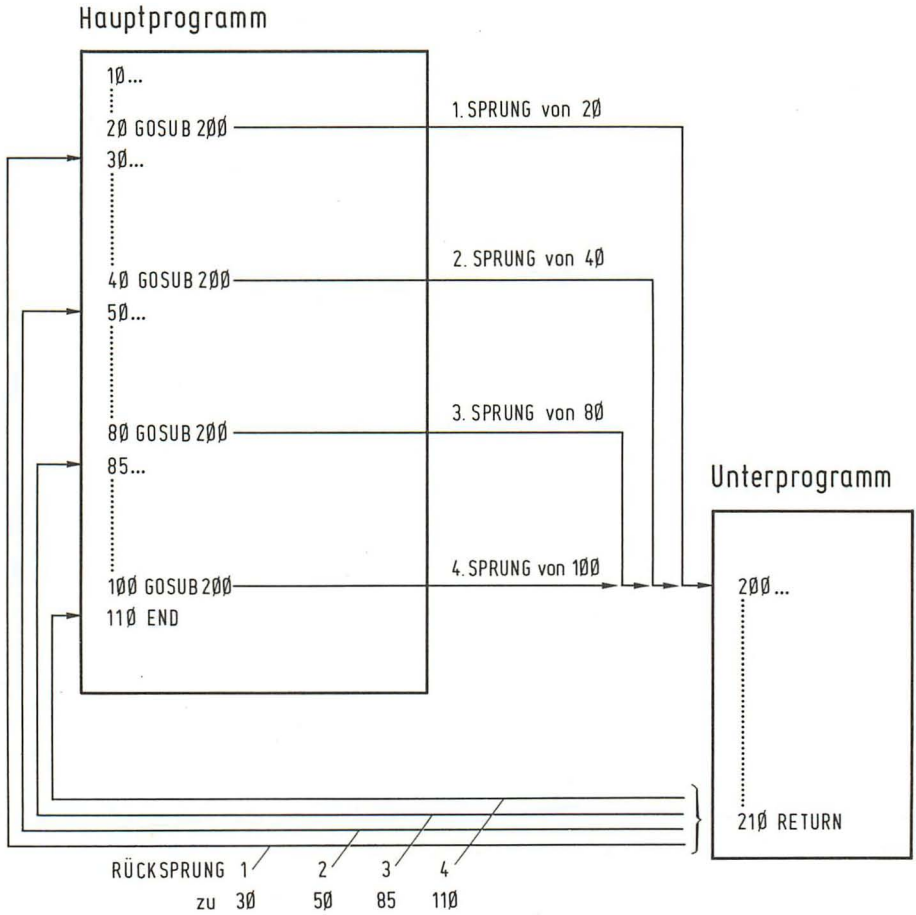


Abb. 18 Von einem Hauptprogramm aus kann ein Unterprogramm von verschiedenen Stellen angesprungen werden. Mit Return findet es automatisch nach Hause ins Hauptprogramm zurück

Sie werden lesen:

IN MUENCHEN STEHT EIN HOFBRAEUHAUS
EINS,ZWEI,GSUFFA!
DA SCHAUT SO MANCHES MADL RAUS
EINS,ZWEI,GSUFFA!

WIE HEISST DER REFRAIN?

EINS,ZWEI,GSUFFA!

WEIL DAS SO SCHOEN IST : NOCH EINMAL !!

EIN,ZWEI,GSUFFA!

Ich denke, Sie haben verstanden, wie das geht? Natürlich können Sie mit GOSUB auch etwas Ernsthaftes betreiben! Hier noch eine Abb. zum Prinzip GOSUB...RETURN (*Abb. 18*).

Was haben Sie dazugelernt?

- ...den Befehl GOSUB – „gehe ins Unterprogramm"!
- Sie müssen nach GOSUB dem Computer ein Ziel geben; sie müssen ihm durch eine Zeilen-Nummer sagen, wo das Unterprogramm beginnt.
- Ob Sie GO TO getrennt- oder GOTO zusammenschreiben, ist dem Computer egal. GOSUB müssen Sie als ein Wort schreiben!
- Ihr mit GOSUB... gestartetes Unterprogramm war im Beispiel nur ein Einzeiler. Sie können das Unterprogramm je nach Erfordernis beliebig lang machen.
- Ist das Unterprogramm zu Ende, müssen Sie es abschließen! Abgeschlossen wird ein Unterprogramm mit RETURN.
- Betrachten Sie die Abb. 18! Dort erfahren Sie, WOHIN der Computer auf den Befehl RETURN zurückkehrt!
  Er geht in die Zeile, die derjenigen folgt, mit der Sie ihn losgeschickt hatten! Im Beispiel hatten Sie ihn in den Zeilen 20, 40, 80 und 100 ins Unterprogramm geschickt. Dieses war mit RETURN korrekt abgeschlossen: der Computer ist in die jeweils folgende Zeile 30, 50, 85, 110 zurückgekehrt.
- Wenn der Computer aus dem Unterprogramm zurück ist, muß er in der betreffenden Zeile vorfinden, wie das Hauptprogramm sinnvoll weitergeht!

31 Lauter Zufälle!

Ob Sie am nächsten Wochenende im Lotto gewinnen, kann Ihnen der Computer leider nicht ausrechnen. Weil das nämlich vom Zufall abhängt. Sicherlich haben Sie schon einmal im Fernsehen die „Ziehung der Lottozahlen" beobachtet? Dort werden aus einem Haufen von 49 Bällen, die mit Zahlen von 1 bis 49 bedruckt sind, durch eine kunstvolle Maschine nacheinander 6 (+1) Bälle herausgefischt. Und wenn Sie genau hinsehen, ist es wirklich ZUFALL, welchen Ball die Maschine erwischt.

Ein anderes Kapitel: Vielleicht haben Sie schon mal – außerhalb unserer Lektionen — mit einem oder gegen einen Computer gespielt? Dann werden Sie vermuten, daß ein Computer auch mit Zufällen arbeiten kann – er kann gewissermaßen auch aus einem „Haufen" von Zahlen eine zufällig herausholen. Dafür gibt es einen speziellen Befehl:

RND (X)*)

RND ist die Abkürzung von RaNDom; Random heißt: ZUFALL.
Wenn der Computer schon nicht ausrechnen kann, mit welchen Zahlen Sie gewinnen – vielleicht kann er Ihnen aus seinem „Zufallshut" ein paar Zahlen für Ihr nächstes Lottospiel zaubern?
Geben Sie ein: **)

```
10 FOR X = 1 TO 7
20 PRINT RND (49)
30 NEXT X
```

Wenn Sie dieses Programm starten, wird Ihnen der Computer 7 zufällig gewählte Zahlen (Zeile 10 !!) hinschreiben, die zwischen 1 und 49 liegen. Siehe Zeile 20 ...RND (**49**)!!
Spielen Sie das einige Male, dann werden Sie zwei Erfahrungen machen:

- Beim zweiten Vorschlag können Zahlen enthalten sein, die Ihnen der Computer schon beim ersten Mal serviert hat. DAS IST ZUFALL!
- Es kann aber auch in EINEM Vorschlag eine Zahl zwei- oder sogar mehrmals „gezogen" werden. DAS IST UNSCHÖN!!

*) Diese Schreibweise gilt für den TRS-80. Prüfen Sie in Ihrem Handbuch, ob das bei *Ihrem* Computer so oder anders geht! Siehe hierzu auch Anhang B, Seite 228.
**) Siehe RND(...); Seite 227, 228.

Offensichtlich arbeitet der Computer nicht wie die Lottozahlenziehmaschine. Die zieht zwar – wie Sie beobachten können – eine Zahl zufällig, legt die gezogene Zahl aber zur Seite! Der Computer aber schmeißt die gezogene Zahl wieder in den Hut! Naheliegend, daß er diese – auch zufällig – nach dem „Mischen" beim nächsten oder übernächsten Mal wieder in der Hand haben kann!

Also: so läßt sich das Ding als Lotto-Spiel-Berater nicht verwenden! Vergessen Sie das Problem für den Augenblick; wir kommen darauf später zurück.

Stattdessen wollen wir mit dem Computer knobeln. Das macht man bekanntlich mit zwei Würfeln, deren Augenzahl sich nach dem Wurf – falls Sie nicht mogeln – auch zufällig ergibt. Mit jedem Wurf kann ein Würfel die Augenzahl 1, 2, 3, 4, 5 oder 6 erreichen. Die Gesamtaugenzahl beider Würfel ist dann mindestens 2 (wenn beide Würfel 1 zeigen) und maximal 12 (wenn beide Würfel 6 zeigen).

Wir machen uns ein kleines Programm:

```
NEW
5 CLS
1Ø A = RND (6) *)
2Ø B = RND (6)
```

Hier ist also A der erste, B der zweite Würfel. Sie lernen „zufällig", daß Sie in *einem* Computerprogramm auch zwei (oder mehrere) Zufallsgeneratoren anwerfen können.

```
3Ø N = A + B
```

Mit dieser Zeile bildet der Computer jeweils die Summe der mit beiden Würfeln A und B gezogenen Augenzahlen.

```
4Ø PRINT"AUGENZAHL",,,"AUGENZAHL"
5Ø PRINT"WUERFEL 1",,,"WUERFEL 2"
6Ø PRINT"==========================================
======"
65 PRINT A,,,B
7Ø PRINT
8Ø PRINT"GESAMT-AUGENZAHL "; N
9Ø FOR X = 1 TO 1ØØØ : NEXT
1ØØ GO TO 5
```

Wenn Sie dieses Programm starten, können Sie eine Weile beobachten, wie der Computer *zufällige* Zahlen zwischen 1 und 6 für jeden Würfel ausgibt. Es kann sein, daß er zum Beispiel dreimal hintereinander für Würfel A die 2 zieht. Das ist

*) Siehe RND(...); Seite 227, 228.

ZUFALL! Sie können schließlich beim Würfeln auch dreimal hintereinander die 2 werfen! Erst wenn Sie das Spiel millionenfach spielen – was ich Ihnen nicht empfehlen möchte – werden Sie erkennen, daß für jeden Würfel alle 6 Zahlen fast gleich oft gezogen werden!

So ist das Spiel natürlich langweilig. Sie sollten mit Ihrem Computer Spielregeln vereinbaren! Wie wär's mit folgenden:

Wenn der Computer zwei „Einsen" hat, soll ER gewinnen! Ferner – wenn der Computer zwei „Sechsen" hat, soll er auch gewinnen. In allen anderen Fällen gewinnen Sie!

Das programmieren wir so:

```
85 IF N = 2 THEN 102
86 IF N = 12 THEN 102
90 GOSUB 200
100 GO TO 5
102 PRINT
104 PRINT
106 PRINT
110 PRINT" I C H  H A B E  G E W O N N E N ! ! "
120 GOSUB 200
130 GO TO 5
200 FOR X = 1 TO 1000 : NEXT
210 RETURN
```

Das ist nicht besonders fair! Der Computer hat viel weniger Chancen als Sie! Verbessern Sie doch einmal die Spielregeln derart zu seinen Gunsten, daß Sie beschließen:

Wenn für N eine gerade Zahl herauskommt, dann haben SIE gewonnen.

Wenn für N eine ungerade Zahl herauskommt, dann hat der Computer gewonnen!

Da sollten Sie natürlich wissen, wie man per Computer eine Zahl daraufhin abklopft, ob sie gerade (das heißt: durch 2 „glatt" teilbar) ist oder nicht! Sie selbst sehen das mit bloßem Auge!

Schieben wir das zunächst einmal ein! Nehmen Sie einige ungerade Zahlen N – z. B. 3 oder 7 oder 9. Dann ist N/2 1.5 oder 3.5 oder 4.5.

Der INT-Teil dieser Zahlen ist dann 1 oder 3 oder 4. Damit ist jeweils der INT-Teil kleiner als N/2 (klar?). Computergerecht könnten Sie definieren:

```
...IF INT(N/2)<N/2 THEN...
```

Sie ändern sinngemäß Ihr Programm:

```
85 X = N/2
```

```
86 IF INT(X)<N/2 THEN 1Ø2
88 PRINT" S I E  H A B E N  G E W O N N E N ! ! "
```

Und damit wünsche ich Ihnen ein paar erholsame Minuten, bevor der Ernst von Basic wieder beginnt!

Was haben Sie dazugelernt?

- RND... ist die Abkürzung von RANDOM; RANDOM heißt Zufall.
- Sie können sich mit dem RND...-Befehl durch den Computer Zahlen *zufällig* „ziehen" lassen.
- Geben Sie ein: RND (Ø), dann liegen die gezogenen Zufallszahlen zwischen Ø und 1, jedenfalls beim TRS-80. Konsultieren Sie Ihr Handbuch, wenn Sie einen anderen Computer besitzen.
- Geben Sie beispielsweise ein RND (1ØØØ), dann liegen die zufällig gezogenen Zahlen zwischen 1 und 1ØØØ einschließlich.
- Beim Arbeiten mit RND... sollten Sie im Kopf haben, daß der Computer jede gezogene Zahl zurück in den Hut wirft. Wo das stört – und es hat uns beispielsweise so gestört, daß wir darauf verzichtet hatten, den Computer als Tipgeber für Lottozahlen einzusetzen – müssen Sie durch eine Programmvariante dafür sorgen, daß eine einmal gezogene Zahl ausgesondert wird! Davon später!
- Sie haben so nebenbei gelernt, wie man eine Zahl daraufhin prüft, ob sie gerade oder ungerade ist.
- Im Beispiel hatten Sie sich für die ungeraden Zahlen X interessiert und folgendes Programm dafür gewählt:
  Erster Schritt:
  Die zu untersuchende Zahl wird durch 2 geteilt.
  Zweiter Schritt:
  Es wird geprüft, ob der INT-Teil von X/2 *kleiner* ist als X/2. Ist das der Fall, dann ist X ungerade.
  Also 1ØØØ Z = X/2
  1Ø1Ø IF INT(Z)<(X/2) THEN...
- Sie hätten natürlich auch untersuchen können, ob die fragliche Zahl eine *gerade* Zahl ist. (Sie kann ja entweder nur das eine oder das andere sein!)
  Dafür gibt es eine elegantere, kürzere Programm-Variante:
  Beispiel: Y = 4 oder 6 oder 12
  Dann ist INT(Y/2) GLEICH Y/2, nämlich 2 oder 3 oder 6!
  Also 1ØØØ IF INT (Y/2) = Y/2 THEN...

32 Der Computer als Kaufmannsgehilfe

Sicherlich werden Sie darauf brennen, nun wieder ein paar nützliche Programme auszuarbeiten. Diese möchte ich jedoch mit einer Vorrede einleiten.

Bei vorangegangenen Programmen hatten Sie oft erlebt, daß die Lösung einer Aufgabe mit einem Taschenrechner schneller – oder mindestens genau so schnell – wie mit dem Computer zu erledigen ist. Sieht man einmal von dem Komfort ab, den der Computer durch seine Fähigkeit bietet, auch Texte zu drucken und Strings zu verarbeiten, dann ist das in der Tat so.

Seine Fähigkeiten entfaltet der Computer am eindrucksvollsten, wenn Sie ein mühsam erarbeitetes Programm *oft* ausnutzen können. Dazu kommt noch etwas anderes: ich gehe davon aus, daß Sie einen Computer besitzen, der für die EINGABE über eine Tastatur, für die AUSGABE über ein Datensichtgerät verfügt. Das aber ist leider nur das halbe Glück und reicht meist für die professionelle oder kommerzielle Anwendung des Computers nicht aus.

Für die Ausgabe wäre ein Drucker nützlich, damit man das Ergebnis der Computermühen schwarz auf weiß besitzt. Und für die Eingabe wäre ein größerer Speicher – für Daten und Programme – erwünscht. Ein Speicher, wie er beispielsweise als „Floppy Disc" zu haben ist. Zwar bieten Personal-Computer die Möglichkeit, Programme auf Cassettenrecordern abzulegen. Aber das ist ein zwar preiswerter, auf die Dauer jedoch wenig befriedigender Behelf. Wir wollen jedoch hier nur das Programmieren einüben – und dazu reicht Ihr Gerät allemal aus. Ich wollte das nur einmal gesagt haben.

32.1 Die Mehrwertsteuer wird erhöht

Seine Schnelligkeit erlaubt es dem Computer, eine große Menge von Daten in kurzer Zeit zu verarbeiten. Eine Preisliste enthält beispielsweise eine Menge Daten; das nächste Beispiel führt den Computer als „Datenverarbeiter" vor.

Nehmen Sie an, Sie haben eine Preisliste mit den Endverbraucherpreisen Ihrer Produkte. Diese enthalten bekanntlich die Mehrwertsteuer – in unserem Falle von 13 %. Und weil Vater Staat die MwSt. auf 14 % erhöht hat, müssen Sie Ihren Kunden eine neue Preisliste zur Verfügung stellen.

Auch dazu geht es zunächst nicht ohne ein wenig Rechnen:

– Wenn A der Preis ist, der 13 % MwSt. enthält...
– dann ist der Ausgangspreis, auf den diese 13 % aufgeschlagen werden: A/1.13

– und der neue Preis – nun mit 14 % MwSt., dann nach Adam Riese (A/1.13)*1.14.

An dieser Stelle zunächst ein Tip: Bei der Anwendung – aber auch bei der computergerechten Aufbereitung – von Formeln können sich Fehler einschleichen, die dann zwangsläufig zu einem falschen Ergebnis führen. Es ist immer sehr nützlich, durch ein einfaches Zahlenbeispiel, das man zur Not im Kopf rechnen kann, Formel und computergerechte Umwandlung auf Richtigkeit zu prüfen. Ich habe das hier gemacht – und empfehle es Ihnen wärmstens zur Nachahmung.

Und zwar habe ich in der Datenliste des folgenden Programms als erstes Datum A = 113 gewählt. A/1.13 ist dann 100 – das ist also der Ausgangspreis, auf den die 13 % MwSt. erhoben waren. Wenn meine angegebenen Formeln richtig sind, dann muß bei einem Aufschlag von 14 % MwSt. 114 herauskommen. Sie können sich gleich davon überzeugen.

Wenn also dieses Kontrollbeispiel richtig ist,
– dann sind die Formeln vermutlich richtig,
– dann ist das Programm richtig.

Also: bauen Sie beim Programmieren leicht nachprüfbare Kontrollen ein! Und nun zum Programm:

```
NEW
20 PRINT"KATALOGPREIS",,"KATALOGPREIS"
30 PRINT"INKL 13% MWST",,"INKL 14%MWST"
40 PRINT" = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
= = = ="
```

Das ist der Kopf unserer Tabelle. Es geht weiter:

```
50 READ A
60 PRINT A,,(A/1.13)*1.14
```

Stop! A ist aus der alten Preisliste sicher auf den Pfennig genau. Aber wenn wir – beispielsweise – 125.73 DM mit (.../1.13)*1.14 behandeln, dann kommt bestimmt etwas mit mehr als zwei Stellen raus! Wozu können wir inzwischen runden?
Wenn wir jetzt den Ausdruck A/1.13)*1.14
mit dem für Runden A=INT(A*100+.5)/100
kombinieren, möchte ich nicht dafür garantieren, ob das jeder von Ihnen richtig hinkriegt.

Für solche Gefahren bietet der Computer eine einfache Hilfe – und auch die sei zur Nachahmung empfohlen: Wir führen eine Zwischengröße (HILFSVARIABLE) ein und sagen:

```
55 X = (A/1.13)*1.14
```

Und nun runden wir einfach X!

```
58 X = INT(X*100+.5)/100
```

Wir müssen noch Zeile 6Ø ändern zu:

```
6Ø PRINT A,,,X
7Ø GO TO 5Ø
```

Jetzt kommt die Preisliste mit den alten Preisen:

```
8Ø DATA 113,15Ø,5Ø,123,1745,275Ø
9Ø DATA 23.5Ø,17.1Ø,.95,33.2Ø,16.7Ø
```

Wenn Sie Lust haben, können Sie die Zeilen 8Ø, 9Ø mit beliebigen Daten überschreiben. Oder Ihr Programm mit den Zeilen 1ØØ...1ØØØ mit weiteren Daten füttern. Uns kommt es nur auf das Prinzip an, wir begnügen uns also mit den paar Daten!

Wollen Sie sich Ihr Programm mal ansehen? Und wollen Sie es mal in Gang setzen?

32.2 Sie beraten Ihre Kunden

Stellen Sie sich vor, Sie wären Angestellter in einem „Do-it-your-self"-Laden, in dem man neben vielem anderem auch Profilbretter für das Vertäfeln von Wänden und Decken kaufen kann. Und Sie sind der Chef der Holzabteilung. Es ergeben sich dann zwei „Problemfelder":

Die Kunden wissen meist nur, wie lang und wie hoch die Wand ist, die sie vertäfeln wollen. Und fragen „Wieviel Holz brauche ich dafür?" Außerdem wollen sie möglichst kein Brett mehr kaufen, als unbedingt nötig. Und sie wollen natürlich wissen, was das kostet. Sie als Anbieter können schon aus Platz- und Kostengründen nicht jedes x-beliebige Längenmaß führen. Nach Ihren Erfahrungen haben Sie sich für die „Lagerlängen" entschieden:

1.5	2	2.5	3	3.5	4	5	Meter

Diese kosten pro Quadratmeter:

10.—	11.—	12.—	13.—	13.50	14.50	16.—	DM

Weil Sie nicht am Ende des Jahres einen Haufen Brennholz übrigbehalten wollen, ist Ihr Geschäftsprinzip:

- Sie schneiden die Bretter nicht auf Länge zu.
- Sie verkaufen die Bretter nur paketweise. Immer 10 Bretter einer Länge sind in einer Kunststoff-Folie verpackt.

Bevor wir für diese Gegebenheiten ein „Kundenberatungsprogramm" aufstellen, wollen wir das Problem noch vertiefen.

Diese Angaben machen die Kunden (*Abb. 19*):

Und so sehen die Bretter im Querschnitt aus (*Abb. 20*).

Sie sehen, daß die Bretter bei der Montage ineinandergesteckt werden müssen.

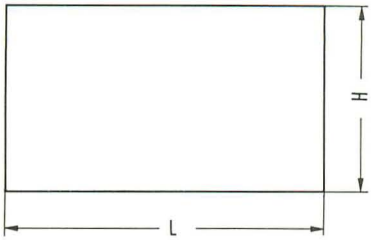


Abb. 19 Wenn Bretter in Längsrichtung des Raumes verlegt werden sollen, dann bestimmt die Höhe des Raumes die Anzahl der nötigen Bretter

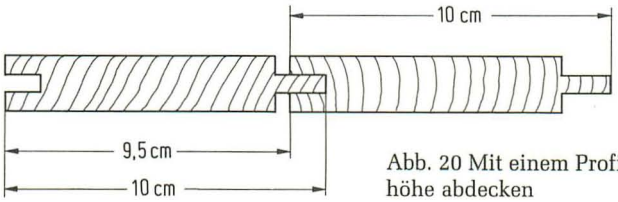


Abb. 20 Mit einem Profilbrett können Sie 9,5 cm Wandhöhe abdecken

Deswegen ist zur Bedeckung der gleichen Fläche mehr Holz nötig, als wenn die Bretter nebeneinander gelegt werden würden.

Sie berücksichtigen diesen Umstand durch einen Zuschlag von 5 % (in Zeile 15). Versuchen Sie zunächst herauszufinden, wieviel Pakete Bretter – unabhängig von der Länge des Raumes – Ihr Kunde benötigt. Dazu müssen Sie von der Höhe des Raumes ausgehen:

```
NEW
5 INPUT"HOEHE DES RAUMES";H
```

Da jedes Brett 10 cm oder .1 m „hoch" ist, ergibt sich die Zahl der Bretter zu:

```
1Ø Z = H/.1
15 Z = Z*1.Ø5
```

Wenn Sie jetzt einmal eine Zwischenrechnung machen, werden Sie z. B. erhalten:

Höhe des Raumes	H:	2	2.25	2.5	Meter
Anzahl der Bretter	Z:	21	23.625	26.25	Stück

Sie verkaufen aber nicht ein Brett, schon gar nicht 3.625 Stück! Also nur 10 oder 20 oder 30; also 1 oder 2 oder 3 Pakete. Sie kommen wie folgt auf die Zahl der Pakete:

- teilen Sie die Zahl der Bretter durch 10
- bilden Sie davon den INT-Teil
- berücksichtigen Sie die „Zehntel"-Pakete, indem Sie einfach eins zuschlagen!

Das ergibt:

```
2Ø Z = INT(Z/1Ø)
3Ø P = Z + 1
```

In dem Beispiel oben erhalten Sie:

Höhe des Raumes	H:	2	2.25	2.5	Meter
Anzahl der Bretter	Z:	21	23.625	26.25	Stück
...Z/1Ø		2.1	2.3625	2.625	
...INT(Z/1Ø)		2	2	2	
Anzahl Pakete P:		3	3	3	

Der Kunde mit dem 2 m hohen Raum kauft also 9 Bretter mehr, als er benötigt. Grundsätze sind Grundsätze! Nun wird es Zeit, daß Sie Ihren Kunden nach der Länge des Raumes fragen.

```
4Ø INPUT"LAENGE DES RAUMES";L
```

Darauf wird Ihnen fast jeder Kunde eine andere Länge nennen. Sie sind aber gebunden an Ihre insgesamt nur 7 Lagergrößen. Sagt der Kunde – beispielsweise – L = 1.2, dann wissen Sie, daß der Kunde die nächstgrößere Länge mit 1.5 m kaufen sollte.

```
5Ø IF L< = 1.5 THEN 15Ø
```

Was < bedeutet, wissen Sie schon: nämlich „kleiner als". Und <= bedeutet „kleiner als ODER gleich". Um uns das Leben nicht zu schwer zu machen, wollen wir unterstellen, daß die Wand des Kunden „gerade" ist. Wenn er am Fußboden 1.5 m mißt, sollen es auch an der Decke 1.5 m sein. Wir können ihm dann bis einschließlich 1.5 m Zimmerlänge ruhigen Gewissens die Bretter mit L = 1.5 m verkaufen.

Das soll ...L<= 1.5... aussagen.

Nun antworten Sie dem Kunden:

```
15Ø PRINT"SIE BENOETIGEN 1.5 M LANGE BRETTER"
155 END
```

Der Kunde will jetzt wissen, was das kostet. Um ihm eine erschöpfende Antwort zu geben – und um uns das Programmieren zu erleichtern – fügen wir ins Programm ein:

```
44 A$ = "SIE BENOETIGEN"
46 B$ = "PAKETE"
48 C$ = "ZUM GESAMTPREIS VON DM"
```

Zeile 15Ø ändern wir noch einmal:

```
15Ø PRINT A$;P;C$;P*1.5*1Ø
155 END
```

Nach dem letzten Semikolon in Zeile 15Ø steht die Formel für die Preisberechnung:

- P = Anzahl der Pakete
- Ein Paket enthält 10 Bretter zu .1 m Breite; das heißt, je lfd./m Brettlänge steckt in einem Paket 1 Quadratmeter Holz.

– 1.5 m war das Paket lang.
– Von den 1.5 m langen Brettern kostet laut Ihrer Preisliste der Quadratmeter genau 10.– DM.
– Alles klar?

Nun kann der Kunde ja auch andere Längen von Ihnen wollen – beispielsweise von 3.2 m. Sie müssen also alle Eingaben für L darauf prüfen, wie L zu Ihren Lagerlängen steht. Das machen Sie wie folgt:

```
60 IF L< = 2 THEN 160
70 IF L< = 2.5 THEN 170
80 IF L< = 3 THEN 180
90 IF L< = 3.5 THEN 190
100 IF L< = 4 THEN 200
110 IF L< = 5 THEN 210
```

Und nun geben Sie die jeweils richtigen Antworten ein. Wissen Sie noch, warum nach jeder Antwort END stehen muß? Wenn nein, dann schauen Sie auf S. 67 nach.

```
160 PRINT A$;P;B$;C$;P*2*11
165 END
170 PRINT A$;P;B$;C$;P*2.5*12
175 END
180 PRINT A$;P;B$;C$;P*3*13
185 END
190 PRINT A$;P;B$;C$;P*3.5*13.5
195 END
200 PRINT A$;P;B$;C$;P*4*14.5
205 END
210 PRINT A$;P;B$;C$;P*5*16
215 END
```

Als guter Kaufmann werden Sie natürlich nicht auf die Kunden verzichten wollen, deren Wände länger als 5 m sind! Diesen Kunden geben Sie folgenden Rat:

```
120 GO TO 220
220 PRINT"LEIDER SIND LAENGEN UEBER 5 M NICHT VORRAETIG"
230 PRINT"BITTE WAEHLEN SIE AUS UNSEREM LAGER"
240 PRINT"LAGERLAENGE", "PREIS/PAKET","PREIS FUER";P;"PAKETE"
250 PRINT" ==========================================
= = = = ="
260 PRINT"      1.5 M      ","      15 DM      ", P*15;"DM"
270 PRINT"      2 M        ","      22 DM      ", P*22;"DM"
280 PRINT"      2.5 M      ","      30 DM      ", P*30;"DM"
```

```
290 PRINT"      3 M      "," 39 DM     ", P*39;"DM"
300 PRINT"      3.5 M    "," 47.25 DM  ", P*47.25;"DM"
310 PRINT"      4 M      "," 58 DM     ", P*58;"DM"
320 PRINT"      5 M      "," 80 DM     ", P*80;"DM"
330 END
```

Glauben Sie nicht auch, daß Sie mit diesem Programm Ihren Kunden einen guten Service bieten?

Was haben Sie dazugelernt?

Noch einmal alle Zeichen für „Relationen" in einer Übersicht:

= heißt GLEICH
<> heißt NICHT GLEICH oder UNGLEICH
< heißt KLEINER ALS
> heißt GRÖSSER ALS
=< oder <= heißt KLEINER ALS ODER GLEICH
=> oder >= heißt GRÖSSER ALS ODER GLEICH

33 Der Computer als Management-Berater

In den folgenden Programmen möchte ich Ihnen den Computer als einen Mitarbeiter vorstellen, der Sie berät oder der für Sie Prognosen wagt.

Das ist ein außerordentlich gefährliches Unternehmen. Nicht etwa, weil Ihr Mitarbeiter dazu nicht qualifiziert wäre. Aber vielleicht sind Sie – sein Herr – für solche Aufgaben nicht qualifiziert genug. Oder – um es freundlicher zu formulieren – vielleicht können Sie Ihrem Computer für eine zutreffende Prognose nicht genügend exakte Daten liefern.

Ich muß Sie an das Beispiel von Seite 42 erinnern, wo ich Ihnen hoffentlich eindrucksvoll genug vor Augen geführt habe, daß eine Computer-AUSGABE nur so gut sein kann, wie die Computer-EINGABE. Wir leben ja in einer Welt, die computergläubig sein soll. Was wohl heißt, daß wir alles glauben – wenn es nur ein Computer ausgespuckt hat. Daß SIE inzwischen nicht mehr zu solchen Einfaltspinseln zählen, hoffe ich stark!

Wie macht man eine Prognose? Man geht vom IST aus, vom HEUTE (...und GESTERN und VORGESTERN); von Daten und Fakten, die man ganz sicher kennt. Das können 3 Fakten sein oder 30. Sodann entwirft man kühne Modelle; man stellt sich vor, wie die Daten von HEUTE sich in der ZUKUNFT verändern KÖNNTEN. (Denn wenn man wüßte, wie sie – die Daten von HEUTE – sich verändern WERDEN, bräuchte man ja keine Prognose!)

Wird die Weltbevölkerung weiter wachsen? Wenn ja, um wieviel % pro Jahr? Linear? Oder exponentiell? Wollen die Leute noch Autos – oder lieber Fahrräder? Kurz: man trifft einige Annahmen über Dinge, die man nicht kennt; die sich dazu noch gegenseitig – aber man weiß nicht genau wie – beeinflussen können...
...entwickelt ein teures Computerprogramm...
und nennt den Mist, den der Computer am Ende ausgibt, dann auch noch WISSENschaft!

Ich habe mich bewußt so rüde ausgedrückt, um Ihnen das – vielleicht noch vorhandene – letzte Fünkchen Computergläubigkeit auszutreiben. Was ich meine, werden Sie gleich am eigenen Leibe erfahren!

33.1 Wann sind Sie pleite?

Wagen wir eine Prognose! Dazu müssen wir – ausgehend von dem, was wir für heute sicher wissen – einige Annahmen treffen. Annahmen darüber, wie sich das,

was wir wissen, in der Zukunft nach bestem Wissen und Gewissen entwickeln wird. Geschwollen ausgedrückt: Wir entwerfen ein „Szenario".
Hier ist *Ihr* Szenario.

1. Sie sind der Boß einer Firma, die eine Ware zum Stückpreis 100 (DM oder $ oder Rubel – das ist hier unerheblich!) herstellt und am Markt anbietet.
2. Auf einem Markt, der im letzten Jahr 100 000 Stück abgenommen hat; Sie hatten daran einen Anteil von 20 %.
3. Die Anlagen Ihrer Firma sind überaltert, weil Ihre Geldgeber schon seit Jahren kein Geld mehr für Rationalisierungsinvestitionen herausgerückt haben.
4. Sie müssen deshalb Kostensteigerungen über den Preis an den Markt weitergeben, obwohl Sie wissen, daß Ihre hochmoderne, gut verdienende Konkurrenz die Preise eine Weile wird halten können.
5. Ihre Marktforscher haben herausgefunden, daß Ihnen jedes % Preiserhöhung einen Verlust an Marktanteil von 2 % „einbringt". Sie sagen Ihnen ferner, daß der Markt seit einiger Zeit um durchschnittlich 3 % pro Jahr gewachsen ist und sagen das auch für die nächsten Jahre voraus.
6. Ihr Betriebsleiter eröffnet Ihnen zu allem Überfluß, daß eine Produktion von 10 000 Stück/Jahr die unterste Grenze dessen ist, was sich noch fertigen läßt, ohne gleich aufgeben zu müssen. Sonst...
7. *Sie entscheiden:* die Preise werden für die nächsten Jahre „maßvoll" um 8 % im Jahr erhöht.

Preisfrage an Ihren Computer: Wann sind Sie pleite? Sie wissen, daß Sie das so nicht fragen können; der Computer braucht ein Programm.

Die für das Programm notwendigen Daten müssen Sie aus der obigen Schilderung 1 bis 7 herausklauben.
Sie finden dort:

Bei 1:	Ihr Preis ist heute 100		P = 1ØØ
Bei 2:	Der Markt war voriges Jahr (!) 100 000 Stück		M = 1ØØØØØ
	Ihr Markt-Anteil ist 20 %		MA = 2Ø
Bei 3:	sehen Sie, wie schlimm es um Ihre Firma steht. Aber „schlimm" kann der Computer nicht verarbeiten.		
Bei 4:	steht für Ihren Computer auch nichts, mit dem er was anfangen könnte.		
Bei 5:	1 % Preiserhöhung ergibt	MA * .98	
	8 % sind dann:	MA * .84	
Bei 6:	sagt Ihnen der Betriebsleiter:		
	WENN unter 10 000 DANN ist es aus...		IF... THEN...
Bei 7:	entscheiden Sie		P = P * 1.Ø8

Bemerken Sie, wie durch die Verdichtung der bekannten Fakten – und Mutmaßungen – die Sie im letzten Abschnitt betrieben hatten, bereits das Gerippe des Programms hindurchschimmert? Offensichtlich muß es Ziel des Programms sein, herauszufinden, wann Ihre Stückzahl unter 10 000 fällt. Sie könnten das so formulieren:

IF S <= 1ØØØØ THEN PRINT "PLEITE"

Um diesen Punkt zu erwischen, müssen Sie aber auch die Stückzahl-*Entwicklung* in den nächsten Jahren verfolgen.

Zu der Stückzahl gibt es nur eine *indirekte* Aussage. Nämlich:
In einem Markt von 100 000 Stück... haben Sie einen Marktanteil von 20 %. Im Kopf ausgerechnet, ergibt das 20 000. Per Computer das Gleiche! Nämlich S = (M * MA) / 1ØØ.
Tabellieren Sie doch einmal alles, was Sie wissen und wie es sich nach den getroffenen Annahmen verändern wird!

	DAS IST HEUTE	SO WIRD ES SEIN...
MARKT	M = 103ØØØ [1])	M = M * 1.Ø3
IHR ANTEIL	MA = 2Ø	MA = MA * .84
IHRE STÜCKZAHL	S = (M * MA)/1ØØ	S = (M*MA)/1ØØ [2])
IHR PREIS	P = 1ØØ	P = P * 1.Ø8

Zu [1]) Fast hereingefallen? Lesen Sie den Text! Der Markt war *voriges Jahr* 100 000! Heute ist er schon 3 % größer!

Zu [2]) Berühmte Fußangel! S errechnen Sie aus M und MA – auch noch im Jahr 2000! Und M und MA haben Sie ja schon verändert! (in den Zeilen darüber!) Mit diesen „neuen" M und MA berechnen Sie S!!

HALT!
Bevor Sie zum Programmieren schreiten: Die Frage, „Wann sind Sie pleite", enthält eine Frage nach der ZEIT!

Sicher wollen Sie nicht wissen, ob am 15. 6. 84 oder am 17. 7. 84. – Aber das Jahr? Oder in wievielen Jahren? ...das wollen Sie schon wissen. Also lassen Sie einen „Jahreszähler" in Ihrem Programm mitlaufen. Und weil es uns interessiert, lassen Sie sich vom Computer alle Werte ausdrucken, die verfügbar werden!

Es geht an:

```
NEW
1Ø N = Ø                    das ist der Jahreszähler!
2Ø M = 1Ø3 ØØØ
3Ø P = 1ØØ
```

```
40 MA = 20
50 S = (M*MA)/100
52 PRINT"STUECKZAHL","PREIS","MARKTANTEIL","MARKT"
54 PRINT" ==========================================
====="
60 PRINT S,P,MA,M
```

Es folgt die Entwicklung ab dem nächsten Jahr!

```
70 N = N + 1
80 M = M*1.03
90 P = P*1.08
100 P = INT(P*100+.5)/100          (Rundung auf 2 Stellen!)
110 MA = MA*.84
120 MA = INT(MA*100+.5)/100        (dito!)
130 S = (M*MA)/100
140 S = INT(S)                     (Was sollte hier „Rundung"? Sie fertigen doch nur „ganze" Stücke, keine Bruchteile!)
150 IF S <= 10000 THEN 170
160 GO TO 60
170 PRINT S,P,MA,M
180 PRINT
190 PRINT"NACH ";N;" JAHREN SIND SIE PLEITE!"
200 END
```

Wenn Sie dieses Programm starten: Hand auf's Herz – hätten Sie gedacht, daß (unter den getroffenen Annahmen) das „Aus" so schnell kommt?

Erinnern Sie sich noch, was Sie auf den Seiten 87...94 zum Thema „Aussteigen aus Schleifen" eingeübt hatten? Hier führe ich Ihnen eine weitere Variante vor, bei:

```
150 IF S <= 10000 THEN...
```

würde der Computer im 6. Umlauf feststellen, daß S = 9982 ist.
Hieße jetzt – wie früher – die Zeile 170:

```
170 END,
```

dann würde der Computer aussteigen und S = 9982 nicht mehr hinschreiben. Hier steht aber:

```
170 PRINT S,P,MA,M...
```

Und diese Zeile steht nur dazu da, um die 6. Zeile noch hinzuschreiben (und danach noch 180, 190 auszuführen).

33.2 Kampf um Marktanteile

Noch ein „Szenario„!

- Zwei Anbieter tummeln sich – neben anderen – am Markt mit einem vergleichbaren Produkt.
- Der Marktführer hat einen Marktanteil von 30 %, sein schärfster Rivale folgt mit 25 %.
- Die Preise beider Anbieter sind zunächst gleich.
- Während der Zweite am Markt sich mit einer jährlichen Preissteigerungsrate von 2 % begnügt, genehmigt sich der Marktführer 2.5 %.
- Wie wirkt sich dieses Verhalten auf die Marktanteile beider Anbieter aus, unter der Annahme, daß für jeden der Anbieter gilt:
  1 % Preissteigerung ergibt einen Verlust an Marktanteil von 1 %.

Natürlich können Sie das Programm für diese Aufgabe nach dem gleichen Muster stricken wie das vorige. Versuchen Sie es allein! Mein Vorschlag unten enthält nur eine spielerische Variante.

```
10 INPUT"WIEVIELE JAHRE WOLLEN SIE DAS BEOBACHTEN ";N
20 A = 0
30 M = 100
40 M1 = (M*30)/100
50 M2 = (M*25)/100
60 PRINT"MARKTANTEIL",,"MARKTANTEIL"
70 PRINT"ANBIETER 1",,"ANBIETER 2"
80 PRINT" ========================================
====="
90 PRINT M1,,M2
100 A = A + 1
110 M1 = M1 * .975
120 M2 = M2 * .98
```

Hier wurde das Runden vergessen! Es wird nachgetragen:

```
115 M1 = INT(M1*100+.5)/100
125 M2 = INT(M2*100+.5)/100
```

Und nun:

```
130 IF A = N THEN 150
140 GO TO 90
150 END
```

Spielen Sie das ein paarmal mit unterschiedlichem N! Auch dabei kommt eine Überraschung heraus! Es dauert immerhin fast 40 Jahre, bis beide Anbieter einen gleichen – wenn auch sehr niedrigen – Marktanteil haben. Haben Sie übrigens

beobachtet, mit welcher Raffinesse ich mich dieses Mal aus dem Programm geschlichen habe?! Ich kann jetzt durch Eingabe von N in Zeile 1Ø bestimmen wann das Programm verlassen werden soll. Nämlich dann, wenn der anfangs auf Ø gesetzte Zähler A den Wert N erreicht hat!
Siehe:

```
13Ø IF A = N...
```

Die Computerei ist doch recht vielseitig!

33.3 Sie planen mit dem Computer Ihre Investitionen

In den beiden letzten Programmen hatte Ihnen der Computer geholfen, einen Blick in die Zukunft zu tun. Sie mußten dabei für die einzugebenden Daten zum Teil von Schätzungen und Vermutungen ausgehen. Damit wurden die Ergebnisse Ihres Computers zwangsläufig abhängig von der Qualität und Treffsicherheit der von Ihnen getroffenen Annahmen.

Bei den folgenden Programmen – bei denen Ihnen der Computer bei Entscheidungen über Investitionen hilft – sind Ihre Eingabedaten jedoch gesicherte Fakten; die Formeln – die „Regeln" – nach denen der Computer sie verarbeitet, wissenschaftlich abgesichert. Sie könnten auch sagen: Wenn Sie bei den letzten Beispielen mit Hilfe des Computers *geschätzt* haben, dann werden Sie jetzt mit seiner Hilfe *rechnen*.

Es ist wichtig, daß Sie diesen Unterschied bei der Beurteilung der beiden Verfahren genau beachten. In beiden Fällen macht Ihnen der Computer ja nur Vorschläge für Ihre Entscheidungen; in beiden Fällen bleiben Sie Herr des Geschehens. Auch hier will ich Ihnen zunächst das Umfeld schildern.

Sie üben eine Tätigkeit aus, bei der Sie Ihren Kunden, Ihren Vertretern und anderen oft Tabellen, Listen und ähnliche Schriftstücke innerhalb des normalen Schriftverkehrs zuschicken. Sie verwenden dazu Kopien, die Ihre Sekretärin im Kopierzentrum Ihres Unternehmens anfertigen läßt. Und deshalb ergibt es sich oft mehrmals in der Woche, daß Ihre Sekretärin ihre Schreibmaschine verläßt, um die Kopien im einige Stockwerke entfernten Kopierzentrum in Auftrag zu geben, auf ihre Anfertigung zu warten und wieder zurück zu wandern. Das scheint Ihnen unrationell zu sein; Sie überlegen, ob Sie Ihrer Sekretärin nicht ein Kopiergerät für ihren Arbeitsplatz beschaffen sollen. Die Frage ist nur: lohnt sich eine solche Anschaffung?

Ich werde Ihnen helfen, darauf eine Antwort zu finden. Offensichtlich kostet die Laufarbeit Ihrer Sekretärin Zeit, und damit Geld, das Sie einsparen bzw. zur Anschaffung des Kopiergerätes verwenden wollen.

Ein befreundeter Betriebswirt hat mir eine Formel verraten, mit der man solche Entscheidungen rechnerisch absichern kann. Hier ist die Formel:

$$T \times L\,(1 + F) = A \times K$$

Ich sollte Ihnen wohl zuerst die Bedeutung der verschiedenen Buchstaben erklären?

A – das ist der Kapitalbetrag, den Sie investieren wollen. (Im Beispiel für das Kopiergerät). Es leuchtet ein, daß die Anschaffung keinen Sinn macht, wenn A am Ende sehr viel größer ist als der Betrag, den Sie durch Entfall der Lauferei einsparen wollen.

T – ist die Zeit (in Stunden pro Jahr), die Ihre Sekretärin durch die Anschaffung einsparen könnte. Sie haben Ihre Sekretärin gebeten, darüber nachzudenken. Sie hat geantwortet, daß wohl 80 Stunden im Jahr für die Lauferei draufgehen.

L – dahinter verbirgt sich, was Ihre Sekretärin in einer Stunde verdient. Sie bezieht zwar ein monatliches Gehalt. Die Personalabteilung hat Ihnen jedoch gesagt, daß Sie mit einem Stundenlohn von 12.80 DM rechnen können.

F – Was Ihre Sekretärin verdient, ist nur ein Teil der Summe, die sie Ihr Unternehmen kostet. Sie erhält ein Urlaubsgeld. Und einen Betrag zu Weihnachten. Ferner „Lohnfortzahlung im Krankheitsfall", usw. Diese nicht direkt im Monatsgehalt erkennbaren Beträge faßt man zu den „Lohn-Nebenkosten" zusammen. F ist nun der Zuschlagsfaktor, der diese Nebenkosten berücksichtigt. In Ihrem Unternehmen – das wissen Sie auch von der Personalabteilung – rechnet man mit F = 0.4.

K – steht für „Kapital-Wiedergewinnungs-Faktor". Dem liegt die Überlegung zugrunde, daß Ihr Unternehmen das einzusetzende Kapital nicht bar da liegen hat, sondern sich auf einer Bank gegen Zinsen leihen muß. Oder: Wenn das Geld bar vorhanden wäre, dann könnte es Ihre Firma ja gegen gute Zinsen anlegen – statt dafür ein Kopiergerät zu kaufen.
Für K gilt die furchterregende Formel:

$$K = \frac{Z\,(1 + Z)^N}{(1 + Z)^N - 1}$$

Die Buchstaben Z und N bedeuten:

Z – ist der „Zinsfuß". Um zu kurzen Formeln zu kommen, hatten wir oft A = A*1.Ø5 geschrieben, wenn wir A um 5 % erhöhen wollten. Schreiben Sie A = (A*1) + (A*.Ø5), erhalten Sie das gleiche *Ergebnis* – nur dürfen Sie

jetzt .Ø5 den Zinsfuß nennen! Im Augenblick ist Geld sehr teuer. Die betriebswirtschaftliche Abteilung Ihrer Firma rechnet mit 12 % Fremdzins und rät Ihnen, in Ihre Rechnung mit dem doppelten Wert einzusteigen. Deshalb verwenden Sie Z = .24

N – schließlich ist die Anzahl der Jahre, die Sie das Kopiergerät nutzen können, bis es ersetzt werden muß. Allgemein können Sie auch sagen: „in N Jahren muß sich die Anschaffung amortisiert haben". Da das Gerät in einem Büroraum stehen und Ihre Sekretärin sehr sorgfältig damit umgehen wird, setzen Sie für N = 5.

Kehren wir zu der Wunderformel zurück:

T x L (1 + F) = A x K

Da Sie A wissen wollen, stellen Sie die Formel nach A um!

$$A = \frac{T \times L\,(1 + F)}{K}$$

Und nun schreiben Sie das computergerecht:

A = (T * L * (1 + F))/K

Wenden wir uns K zu:

$$K = \frac{Z\,(1 + Z)^N}{(1 + Z)^N - 1}$$

Darin kommt $(1 + Z)^N$ zweimal vor. Sie führen – wie oft geübt – eine Hilfsvariable ein und sagen:

B = (1 + Z)↑N

Dann wird K zu:

K = (Z * B) / (B – 1)

Nun sind Sie in der Lage, zur Lösung Ihres Problems ein Programm für Ihren Computer zu schreiben. Sie verwenden dazu alle Ihnen bekannten Größen wie folgt:

```
NEW
1Ø T = 8Ø
2Ø L = 12.8
3Ø F = .4
4Ø Z = .24
5Ø N = 5
6Ø B = ( 1 + Z ) ↑ N
7Ø K = ( Z * B ) / ( B – 1 )
8Ø A = ( T * L * ( 1 + F ))/ K
```

und jetzt noch: 9Ø PRINT A

Sie haben es jetzt schwarz auf weiß: 3935.78 DM dürfen Sie für das Kopiergerät ausgeben!
Vielleicht sind Sie jetzt neugierig geworden und wollen einmal sehen, welches Ergebnis herauskommt, wenn Sie einzelne Größen variieren? Dann variieren Sie doch einmal N (alles andere bleibt unverändert).

```
2 PRINT"N =",,,"A ="
4 PRINT" ==========================================
====="
50 FOR N = 1 TO 10
85 A = INT(A*100+.5)/100
90 PRINT N,,,A
100 NEXT N
110 END
```

```
N =                                                    A =
= = = =
```

Stunde zahlen und haben außerdem hohe Lohn-Nebenkosten (F = 0.6). Setzen Sie das einmal in das obige Programm ein!

```
10 T = 3024
20 L = 15
30 F = .6
```

N =	A =
1	58529.1
2	105730
3	143795
4	174493
5	199249
6	219214
7	235314
8	248299
9	258770
10	267214

Sind das nicht stattliche Summen? Übrigens: man rechnet heute mit Amortisationszeiten von zwei bis fünf Jahren.

Weil Sie ein derartiges Programm vielleicht in Ihrem Beruf einsetzen können, wollen wir es gemeinsam mal „allgemein gültig" umschreiben!

```
NEW
10 INPUT" WIEVIEL STUNDEN WERDEN PRO JAHR EINGESPART ";T
20 INPUT" WELCHE LOHNKOSTEN IN DM/STD LIEGEN VOR ";L
30 INPUT" WIE IST DER FAKTOR (0.3...0.6) FUER L.-NEBENKOSTEN";F
40 INPUT" WELCHE ZINSEN ZAHLEN SIE ";P
50 Z = ( 2 * P )/100
60 INPUT" IN WIEVIEL JAHREN WOLLEN SIE AMORTISIEREN ";N
70 B = ( 1 + Z ) ↑ N
80 K = ( 2 * B ) / ( B - 1 )
90 A = ( T*L*(1 + F))/K
```

Und das wird die Ausgabe!

```
100 PRINT"BEI ";T;"EINGESPARTEN STUNDEN/JAHR"
110 PRINT"UND ";L;"DM LOHNKOSTEN/STUNDE"
120 PRINT"SOWIE ";N;"JAHREN AMORTISATIONSZEIT"
130 PRINT
140 PRINT"KOENNEN SIE BIS ZU                    ";A;" DM INVESTIEREN"
150 PRINT"                    ==========================="
160 END
```

Bitte beachten Sie die Ausführung der Zeilen 40 und 50!!

34 Der Computer als Textautomat

Sie wissen, was ein String ist und haben davon schon gelegentlich Gebrauch gemacht. Was ist ein Text, ein Brief oder diese Seite anderes, als – im Sinne des Computers – eine Aneinanderreihung von Strings?

Im kommerziellen Einsatz ist der Computer sehr stark damit beschäftigt, Texte zu bearbeiten und zu verarbeiten. Wenn ich Ihnen den Computer jetzt bei dieser Tätigkeit vorführe, dann werden Sie ein weiteres Mal das Fehlen eines Druckers beklagen. Es macht ja wirklich wenig Sinn, verarbeitete Texte nur auf dem Bildschirm zu betrachten. Aber vereinbarungsgemäß wollen wir Basic einüben – und das geht auch ohne Drucker.

Sie möchten Ihrer Tante Berta zum Geburtstag gratulieren. Sie setzen sich hin und schreiben etwa:

Liebe Tante Berta!
Ich gratuliere Dir herzlich zum Geburtstag.
Dein Neffe Klaus.

Als Computerbesitzer wollen Sie diesen Vorgang des Brief-Schreibens rationalisieren. Sie wollen die Arbeit durch Ihren Computer erledigen lassen. Sie schreiben ein Programm!

```
10 A$ = "LIEBE TANTE BERTA!"
20 B$ = " ICH GRATULIERE DIR HERZLICH ZUM GEBURTSTAG."
30 C$ = " DEIN NEFFE KLAUS."
```

Ich habe hier aus Gründen der Übung – und weil ich nicht weiß, wieviel Text Ihr Computer in einem String schluckt – verhältnismäßig kurze Strings gewählt. Der TRS-80 hätte den ganzen Brief auch in einem String aufgenommen. Sie müssen das halt ausprobieren; wir haben schon einmal darüber gesprochen.

Sie haben also jetzt den Brief (Text) in Ihrem Computer gespeichert. Wenn das Ereignis naht, sagen Sie Ihrem Computer:

```
40 PRINT A$;B$;C$
```

Der Computer wird ausgeben:

```
LIEBE TANTE BERTA !ICH GRATULIERE DIR HERZLICH ZUM GEBURTSTAG.D
EIN NEFFE KLAUS.
```

Ihre Tante wird über so einen Brief nicht begeistert sein. Das hängt alles so lieblos aneinander; außerdem: Seit wann kann man das Wort DEIN trennen? Und dann: D EIN??

Ich habe hier – auch zur Übung – angenommen, daß mit dem „D" in dem Wort DEIN eine Zeile Ihres Bildschirmes voll ist; der Computer schreibt dann ohne Rücksicht auf Grammatik den Rest auf die nächste Zeile.

Wir können den Computerbrief sehr einfach verschönern!

```
40 PRINT A$     |   60 PRINT B$
50 PRINT        |   70 PRINT C$
```

Jetzt sieht der Brief so aus:

LIEBE TANTE BERTA!
ICH GRATULIERE DIR HERZLICH ZUM GEBURTSTAG.
DEIN NEFFE KLAUS.

Und so wollen wir ihn gelten lassen. Rechte Befriedigung werden Sie sicher trotzdem nicht empfinden. Variieren Sie doch die Aufgabe wie folgt:

- Sie wollen Tante Berta zum GEBURTSTAG und zum HOCHZEITSTAG gratulieren.
- Außerdem wollen Sie jeweils angeben, um den wievielten Jubeltag es sich handelt.

Ergänzen Sie Ihr Programm zunächst um:

```
20 B$ ="ICH GRATULIERE DIR ZUM"
22 D$ ="GEBURTSTAG"
24 E$ ="HOCHZEITSTAG"
```

Jetzt fehlt noch die Eingabe des „Datums"! Das Wort EINGABE weist Ihnen den Weg zu:

```
35 INPUT"DER WIEVIELTE GEBURTS- ODER HOCHZEITSTAG";N$
```

Nun können Sie programmieren:

1. Gratulation zum Geburtstag:
```
40 PRINT A$
50 PRINT
60 PRINT B$;N$;D$          D$ für Geburtstag!
70 PRINT C$
```
2. Gratulation zum Hochzeitstag:
```
40 PRINT A$
50 PRINT
60 PRINT B$;N$;E$          E$ für Hochzeitstag!
70 PRINT C$
```

Wenn Sie diese Programme starten, findet der Computer zunächst lauter Zuweisungen (Zeilen 10, 20, 22, 24). In Zeile 35 hält er an, um auf Ihre Eingabe für das „Datum" zu warten. Auch das macht noch nicht viel Sinn. Wir wenden uns einem anderen Beispiel zu.

Sie betreiben einen Versandhandel und bekommen per Post Aufträge zugeschickt. Sie bedanken sich bei Ihren Kunden; je nach der vorliegenden Situation sind drei Brieftexte denkbar:

Brief 1 (Auftragsbestätigung):

Sehr geehrter Herr [Müller !]
Wir danken Ihnen für Ihren Auftrag vom [17. Mai] .
Ihren Auftrag können wir sofort ausführen.
Wir werden die Ware hier am [18. Mai] abschicken.
Wir danken Ihnen und verbleiben
mit freundlichen Grüßen.
Meier & Co

Brief 2 (Terminangabe):

Sehr geehrter Herr [Müller] !
Wir danken Ihnen für Ihren Auftrag vom [17. Mai] .
Leider ist der bestellte Artikel im Augenblick nicht lieferbar. Voraussichtlich wird die Bestellung am [15. August] ausgeführt.
Wir danken Ihnen und verbleiben
mit freundlichen Grüßen
Meier & Co

Brief 3 (Neuangebot):

Sehr geehrter Herr [Müller] !
Wir danken Ihnen für Ihren Auftrag vom [17. Mai] .
Der bestellte Artikel ist leider ausverkauft.
Wir können Ihnen jedoch das Nachfolgemodell anbieten. Bitte lassen Sie uns wissen, ob Sie damit einverstanden sind.
Wir danken Ihnen und verbleiben
mit freundlichen Grüßen
Meier & Co

Nun achten Sie darauf, was für Zeichen ich in den einzelnen Briefen angebracht habe.

[ ] Alle eingerahmten Angaben müssen wohl in jedem Brief verschieden sein.
- Die Kunden haben verschiedene Namen.
- Sie bestellen zu verschiedenen Terminen.
- Sie liefern zu unterschiedlichen Terminen.

Sehr... Alles, was unterstrichen ist, das ist in allen drei Briefen gleich.

Der restliche Text ist zwar von Brief zu Brief verschieden, kann aber bei mehreren Briefen der gleichen Art (z. B. Auftragsbestätigung) ebenfalls immer gleich sein. Zur Programmierung zerlegen Sie den Brief in einzelne Strings. Alles, was „individuell" ist, wird darüber hinaus über INPUT eingegeben. Dieser Teil der Programmierung führt zu folgendem Ergebnis:

```
  5 CLS
 10 A$  = "SEHR GEEHRTER HERR"
 20 B$  = "WIR DANKEN IHNEN FUER IHREN AUFTRAG VOM"
 30 C$  = "IHREN AUFTRAG KOENNEN WIR SOFORT AUSFUEHREN."
 40 D$  = "WIR WERDEN DIE WARE HIER AM"
 50 E$  = "ABSCHICKEN."
 60 F$  = "WIR DANKEN IHNEN UND VERBLEIBEN"
 70 G$  = "MIT FREUNDLICHEN GRUESSEN"
 80 H$  = "MEIER & CO"
 90 I$  = "LEIDER IST DER BESTELLTE ARTIKEL IM AUGENBLICK NICHT
          LIEFERBAR"
100 J$  = "VORAUSSICHTLICH WIRD DIE BESTELLUNG AM"
110 K$  = "AUSGEFUEHRT."
120 L$  = "DER BESTELLTE ARTIKEL IST LEIDER AUSVERKAUFT."
130 M$  = "WIR KOENNEN IHNEN JEDOCH EIN NACHFOLGEMODELL AN-
          BIETEN."
140 N$  = "BITTE LASSEN SIE UNS WISSEN, OB SIE DAMIT EINVERSTAN-
          DEN SIND."
150 INPUT "NAME DES KUNDEN";O$
160 INPUT "STRASSE, HAUSNUMMER";P$
170 INPUT "WOHNORT";Q$
180 INPUT "BESTELL-DATUM"; R$
190 INPUT "LIEFERDATUM";S$
```

Um nun jeweils den richtigen Brief vom Computer schreiben zu lassen, machen wir zunächst ein Programm mit einer Auswahltabelle. Das geschieht in den Zeilen 200 bis 280. In dem Programmteil ab Zeile 300 werden dann die einzelnen Strings wieder zu den gewünschten Briefen zusammengesetzt.

```
200 CLS
210 PRINT "     AUSWAHLTABELLE"
220 PRINT "     ============"
230 PRINT
240 PRINT "AUFTRAGSBESTAETIGUNG = 1"
250 PRINT "TERMINANGABE         = 2"
260 PRINT "NEUANGEBOT           = 3"
270 INPUT A
```

```
280 ON A GO TO 300, 450, 600
300 CLS
310 PRINT O$
320 PRINT P$
330 PRINT Q$
340 PRINT
350 PRINT A$; O$; "!"
360 PRINT
370 PRINT B$; R$; "."
380 PRINT C$
390 PRINT D$; S$; E$
400 PRINT
405 PRINT F$
410 PRINT G$
420 PRINT
430 PRINT H$
440 END
450 CLS
460 PRINT O$
470 PRINT P$
480 PRINT Q$
490 PRINT
500 PRINT A$; O$; "!"
510 PRINT
520 PRINT B$; R$; "."
530 PRINT I$; "."
540 PRINT J$; S$; K$
545 PRINT
550 PRINT F$
560 PRINT G$
570 PRINT
580 PRINT H$
590 END
```

BRIEF 1

BRIEF 2

```
600 CLS
610 PRINT O$
620 PRINT P$
630 PRINT Q$
640 PRINT
650 PRINT A$; O$; "!"
660 PRINT
670 PRINT B$; R$; "."
680 PRINT L$
690 PRINT M$
700 PRINT N$
710 PRINT
720 PRINT F$
730 PRINT G$
740 PRINT
750 PRINT H$
760 END
```

BRIEF 3

Auf ein paar Besonderheiten möchte ich Sie noch hinweisen: Betrachten Sie die Zeile 500:

```
500 PRINT A$;O$;"!"
```

A$ ist: SEHR GEEHRTER HERR; O$ ist: SCHULZE – beispielsweise! In Zeile 500 wird daraus:

SEHR GEEHRTER HERR SCHULZE!

Bemerken Sie, wie das Ausrufungszeichen in den Brief gelangt ist? Wenn sich also – wie hier – Satzzeichen aus bestimmten Gründen im String nicht unterbringen lassen, dann werden Sie bei der Ausführung auf die bekannte Weise (durch ".....") dem Computer zum Druck angewiesen!

So hundertprozentig gefällt mir das Programm noch nicht! In den einzelnen Unterabschnitten kommen mir zu oft die gleichen PRINT-Befehle vor. Warum machen wir nicht aus Unterabschnitten ein UNTERPROGRAMM?

Sie haben doch gelernt, wie das mit GOSUB...RETURN geht! Ich führe Ihnen das noch einmal vor!

```
 5 CLS
10 A$ = "SEHR GEEHRTER HERR"
20 B$ = "WIR DANKEN IHNEN FÜR IHREN AUFTRAG VOM"
30 C$ = "IHREN AUFTRAG KOENNEN WIR SOFORT AUSFUEHREN."
40 D$ = "WIR WERDEN DIE WARE HIER AM "
50 E$ = "ABSCHICKEN."
60 F$ = "WIR DANKEN IHNEN UND VERBLEIBEN"
70 G$ = "MIT FREUNDLICHEN GRUESSEN"
```

```
 80 H$  = "MEIER & CO"
 90 I$  = "LEIDER IST DER BESTELLTE ARTIKEL IM AUGENBLICK
          NICHT LIEFERBAR"
100 J$  = "VORAUSSICHTLICH WIRD DIE BESTELLUNG AM "
110 K$  = "AUSGEFUEHRT."
120 L$  = "DER BESTELLTE ARTIKEL IST LEIDER AUSVERKAUFT."
130 M$  = "WIR KOENNEN IHNEN JEDOCH EIN NACHFOLGEMODELL
          ANBIETEN."
140 N$  = "BITTE LASSEN SIE UNS WISSEN, OB SIE DAMIT
          EINVERSTANDEN SIND."
150 INPUT "NAME DES KUNDEN"; O$
160 INPUT "STRASSE, HAUSNUMMER"; P$
170 INPUT "WOHNORT"; Q$
180 INPUT "BESTELL-DATUM"; R$
190 INPUT "LIEFERDATUM"; S$
```

Sie sehen, daß „Zuweisungsteil" und INPUT-Teil unverändert bleiben. Auch die Auswahltabelle bleibt mit den Zeilen 200 bis 280 erhalten.

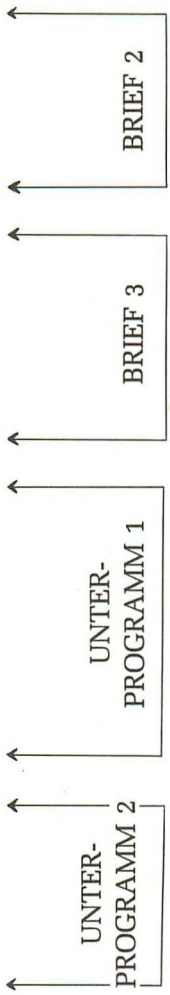
Aber ab Zeile 300 gehen wir in das Unterprogramm, das mit Zeile 700 beginnt. Dieses Unterprogramm druckt Name, Straße und Wohnort des Kunden. Dann die Anrede und das „Dankeschön" für den Auftrag. Dann kehrt es zurück ins Hauptprogramm (mit RETURN!) und findet dort – beispielsweise – in Zeile 310 den weiteren Text.

Ein zweites Unterprogramm beginnt – z. B. – in Zeile 330: GOSUB 800. In diesem Unterprogramm befindet sich die für jeden Brief gleiche „Schlußformel".

```
200 CLS
210 PRINT "  AUSWAHLTABELLE"
220 PRINT "  ============"
230 PRINT
240 PRINT "AUFTRAGSBESTAETIGUNG = 1"
250 PRINT "TERMINANGABE         = 2"
260 PRINT "NEUANGEBOT           = 3"
270 INPUT A
280 ON A GO TO 300, 450, 600
300 GOSUB 700
310 PRINT C$
320 PRINT D$; S$; E$

330 GOSUB 800
340 END
```

BRIEF 1

```
450 GOSUB 700
460 PRINT I$;"."
470 PRINT J$; S$; K$

480 GOSUB 800
490 END
600 GOSUB 700
610 PRINT L$
620 PRINT M$
630 PRINT N$

640 GOSUB 800
650 END
700 CLS
710 PRINT O$
720 PRINT P$
730 PRINT Q$
740 PRINT
750 PRINT A$; O$; "!"
760 PRINT
770 PRINT B$; R$; "."
780 RETURN
800 PRINT
810 PRINT F$
820 PRINT G$
830 PRINT
840 PRINT H$
850 RETURN
```



Was haben Sie dazugelernt?

- ...Wie man Texte in Strings umwandelt.
- ...Wie man aus Strings wieder Texte zusammensetzt.
- Lassen sich aus besonderen Gründen Satzzeichen, Leerzeilen „individuelle" Daten usw. nicht in einem String unterbringen, dann kann man sie über PRINT "....." einfügen.
- Mit INPUT... kann man nicht nur – wie in unseren Beispielen – besondere Strings ins Programm aufnehmen. Sie können auch INPUT-Befehle mitten im Text anordnen. Der Computer „druckt" dann den Text bis zum INPUT-Befehl, wartet auf die EINGABE und setzt danach den „Druck" fort. Das ist aber wirklich nur sinnvoll, wenn auch ein Drucker zur Verfügung steht!

35 Lagertechnik

Stellen Sie sich vor, Sie seien Besitzer eines Schuhgeschäftes. Sie führen Schuhe für Damen, Herren und Kinder. Das sind drei Angebote. Für jede Kundengruppe führen Sie Straßenschuhe, Hausschuhe, Turnschuhe und Wanderschuhe. Das sind schon 3 x 4 = 12 Angebote. Da die Geschmäcker bekanntlich verschieden sind, führen Sie Ihr Angebot noch in 5 verschiedenen, modischen Varianten und in drei Farben. Dann haben Sie 3 x 4 x 5 x 3 = 180 Angebote.

Die Füße Ihrer Kunden sind natürlich unterschiedlich groß – also nehmen Sie alle Modelle noch in acht Größen an Lager – darin verbergen sich dann 180 x 8 = 1440 verschiedene Schuhe, wenn Sie jedes Paar nur einmal am Lager haben...

Wir wollen dieses Spiel hier abbrechen. Sie haben längst erkannt, worauf das hinausläuft. Schon bei einem vergleichsweise bescheidenen Angebot kommen Sie rasch zu einer Fülle von VARIANTEN. Automobilfabriken lagern z. B. in ihren Ersatzteillagern mehr als 100 000 verschiedene Teile – von der Unterlegscheibe bis zum Austauschmotor. Es ist leicht einzusehen, daß Sie in einem derartigen Lager niemals das gewünschte Teil finden würden, wenn das Lager nicht gut organisiert wäre und jedes Ding seinen festen Platz hätte.

Wenn Sie als Schuhkaufmann Ihren Lehrling mit dem Auftrag ins Lager schicken: „Hole die roten Damenturnschuhe mit Ledersohle...", dann werden Sie ihn eine Weile nicht wiedersehen. Sie müßten ihm fairerweise sagen, WO er die betreffenden Schuhe findet, WO im LAGER die Schuhe nach der Anlieferung EINGELAGERT wurden. Und Sie müssen – wenn Sie nicht ein Chaos produzieren wollen – im voraus für *bestimmte* Waren *bestimmte* (Lager-)PLÄTZE festlegen.

Sie können das in epischer Breite tun, wie „drittes Fach in der siebten Reihe des zweiten Regals".
Oder in militärischer Kürze:

020703; wobei jetzt steht:
02 für zweites Regal
07 für siebte Reihe
03 für drittes Fach.
Wir wollen unseren Computer nicht aus den Augen verlieren. Aus dem oben Erzählten ziehen wir die wichtigsten Begriffe heraus und versuchen, sie in einen Bezug zum Computer zu bringen.

Typen-Vielfalt, Varianten

Das klingt doch fast wie VARIABLE – und damit haben Sie umzugehen gelernt.

Sie könnten auf den Gedanken kommen, die einzelnen „Schuh-Variablen" mit Namen zu belegen wie A, A9, AA...AZ usw.

Wenn Sie in die Variablen-Liste auf Seite 47 zurückgehen, werden Sie erkennen, daß Sie dort eine Menge Variable zur Verfügung haben. Sie erkennen aber auch, daß die Zahl der möglichen Variablen für einige Tausend „Lagergüter" NICHT AUSREICHT.

Wir führen einen neuen Variablen-Typ ein – die sogenannte
INDIZIERTE VARIABLE
Übersetzen Sie das fürs Gedächtnis zu „Variable mit Index".

In der Mathematik schreibt man einen Index so:
X_1, gesprochen „X-eins" oder
X_{29}, gesprochen „X-neunundzwanzig".
Sie sehen hier schon: In der Variablenliste gibt es keine Variable mit dem Namen X 29...

Der Computer akzeptiert aber nicht X_{29}. Wenn Sie also der Variablen X den Index 29 geben wollen, müssen Sie das für den Computer verständlich machen. Und zwar so:
X_{29} schreibt man für den Computer: X(29)

LAGER
...das war ein weiterer Begriff. Das setzen Sie gleich mit dem SPEICHER Ihres Computers.

Lager einrichten

Wenn Sie 1000 Artikel lagern wollen, dann werden Sie kein Lager bauen, das nur für 500 Artikel ausreicht. Aber auch keins für 10 000, weil das viel zu teuer und damit unwirtschaftlich wäre.

Bei Ihrem Computer richten Sie ein Lager ein...
DIMensionieren Sie einen bestimmten SPEICHERbereich durch den Befehl: DIM...
Das übersetzen Sie zu DIMensioniere!
...richte ein!
...schaffe Platz für!...

LAGER EINRÄUMEN
Das müssen Sie per Computer tun; Sie müssen den SPEICHER mit DATEN – z. B. mit indizierten Variablen FÜLLEN.
AUS DEM LAGER SUCHEN / LAGER AUSRÄUMEN

Auch das tut der Computer für Sie – natürlich nur auf den entsprechenden Befehl! Wie stets, wollen wir das an einem einfachen Beispiel üben.

1. Schritt: Lager einrichten

```
10 DIM A(8)
```

Das bedeutet: der Computer soll für 8 (!) indizierte Variable im Speicher Platz schaffen und freihalten.

Sinnvoller und eindrucksvoller wäre natürlich DIM A(600) – aber dann bekämen Sie beim anschließenden „Einräumen" lange Arme...

2. Schritt: Lager einräumen

Sie müssen nun die DIMensionierten 8 Speicherplätze mit 8 Werten belegen.

```
20 FOR I = 1 TO 8          (8 Plätze!!)
30 INPUT A(I)
40 NEXT I
```

Für das in Zeile 10 DIMensionierte Lager bedeuten die Zeilen 20, 30 und 40 den „Einräum- oder Einlesevorgang".

Es ist gleichgültig, ob Sie schreiben

```
...FOR I...          oder          ...FOR X...
...A(I)                            ...A(X)
NEXT I                             NEXT X
```

Bevor wir „einräumen", wollen wir den 3. Schritt programmieren.

3. Schritt: Lager ausräumen

```
45 PRINT"A(7)=","A(1)=","A(5)=","A(2)="
50 PRINT A(7),A(1),A(5),A(2)
```

Hier wollen wir mit Zeile 50 nicht alle 8 Variablen sehen, sondern nur die mit dem Index 7, 1, 5, 2.

Wenn Sie jetzt das Programm starten, wird der Computer achtmal nach A(...) fragen. Wir schreiben das einmal mit!

Ihre Eingabe für A(...)

```
? 17        für A(1)
? 25        für A(2)
? 33        für A(3)
? 44        für A(4)
? 125       für A(5)
? 32        für A(6)
? 15        für A(7)
? 8         für A(8)
```

Achtung! Wenn Sie jetzt ENTER drücken, geht der Computer sofort im Programm weiter; von Zeile 40 in Zeilen 45, 50. Sie lesen:

A(7)=	A(1)=	A(5)=	A(2)=
15	17	125	25

Die Kontrolle mit Ihrer Eingabe oben ergibt, daß der Computer *richtig* ein- und ausgelesen hat! Machen wir noch ein Beispiel:

```
NEW
10 DIM A(6)
20 FOR Z = 1 TO 6
30 INPUT A(Z)
40 NEXT Z
```

Als Variante unseres Programms wollen wir einmal die Summe S für alle 6 eingegebenen Variablen ausrechnen lassen!

```
50 S = 0
60 FOR Z = 1 TO 6
70 S = S + A(Z)                    (Hiermit erfolgt die Summenbildung!)
80 PRINT A(Z)
90 NEXT Z
100 PRINT"SUMME ALLER A(Z) =";S
```

Es geht los!

```
? 1
? 2
? 3        EINGABE!
? 4
? 5
? 6

1
2
3          AUSGABE!
4
5
6

SUMME ALLER A(Z)=21
```

Unser bisheriges Lager hatte nur eine DIMension. Alle Lagerfächer waren gewissermaßen in eine Reihe nebeneinander gestellt worden.

Das ist nicht sonderlich praktisch; angesichts der hohen Preise für Grund und Boden geht man heute beim Bau von Lagern in die Höhe!

Schauen Sie sich die *Abb. 21* an:
Dort ist ein „Lager" mit 6 „Stockwerken", 4 „Reihen" und insgesamt 24 Plätzen abgebildet. Die Plätze sind von A(1) bis A(24) numeriert.

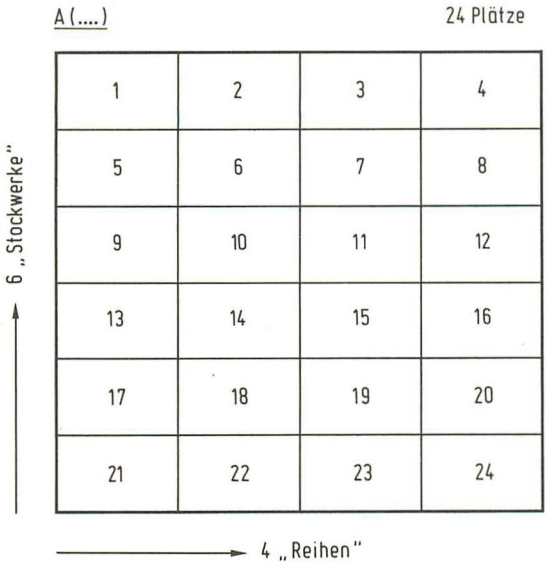


Abb. 21 Ein Lager kann zwei (und auch mehr) Dimensionen annehmen

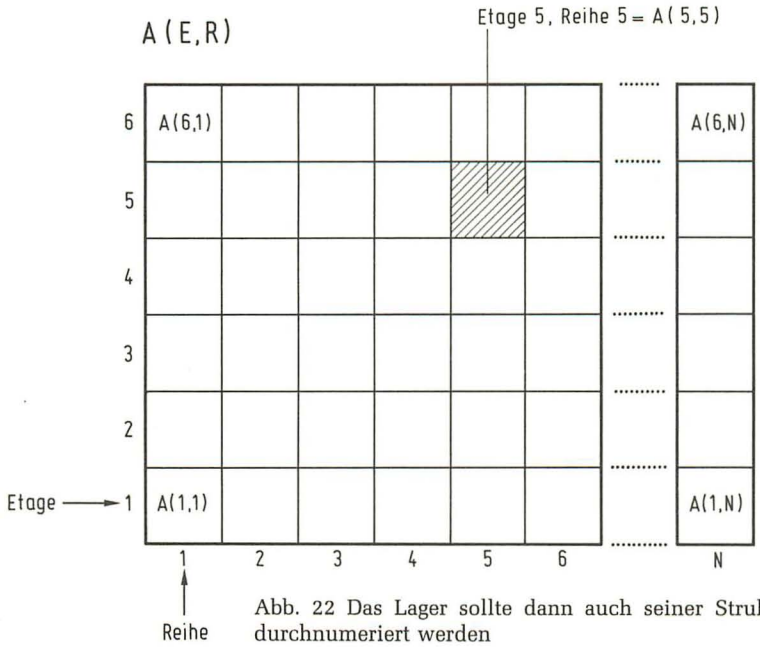


Abb. 22 Das Lager sollte dann auch seiner Struktur entsprechend durchnumeriert werden

So etwas kann man machen – aber es ist nicht besonders übersichtlich. Schauen Sie sich zum Vergleich die *Abb. 22* an!

Jetzt ist hinsichtlich der Plätze eine *zweifache* Unterscheidungsmöglichkeit getroffen.

Nämlich nach	Etagen	(im Beispiel 6)
und nach	Reihen	(im Beispiel N)

In die Computersprache übersetzt, ergibt das den Index E, R für die einzelnen Fächer.
A(E,R)
Die Abbildung macht wohl deutlich, wie das zu lesen und zu verstehen ist!

35.1 Der Computer verwaltet Ihr Scheckbuch

Diese „Hochregallagertechnik" wollen wir an einem praktischen Beispiel üben. Als ordentlicher Mensch schreiben Sie sich auf, welche Schecks Sie ausstellen. Diese LISTE enthält die Scheck-Nummer, das Ausstell-Datum und den Scheck-Betrag – etwa nach folgendem Schema:
(Wir begnügen uns mit einer Liste für 10 Schecks!) Mit den Daten Ihrer Liste sieht das so aus:

1. Schritt: „Lager" einrichten:
10 DIM SB (1Ø, 3)
Für SB = Scheck-Buch mit 10 Zeilen und 3 Spalten

2. Schritt: „Lager" einräumen:
Das machen wir mit zwei ineinandergeschalteten FOR...NEXT-Schleifen. 4Ø READ... weist darauf hin, daß wir die Daten der Schecks als DATA-Liste eingeben. Das hat folgenden Vorteil: Bei den Übungen vorhin war aufgefallen, daß die über INPUT eingegebenen Variablen nach jedem Computerlauf automatisch gelöscht wurden. Bei READ... DATA... sind die Daten jedoch Bestandteil des Programms und bleiben solange erhalten und „verarbeitbar", wie das Programm existiert.

```
2Ø FOR Z = 1 TO 1Ø
3Ø FOR S = 1 TO 3
4Ø READ SB (Z, S)
5Ø NEXT S
6Ø NEXT Z
```

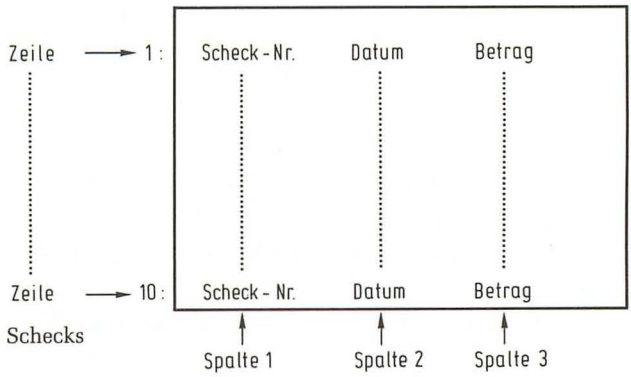


Abb. 23 Die Liste der ausgestellten Schecks ist hier zweidimensional

	"Scheck-Nr."	"Datum"	"Betrag"
	Spalte 1	Spalte 2	Spalte 3
	S = 1	S = 2	S = 3
Z = 1	19	1.12	115
2	2Ø	2.12	125.3Ø
3	21	5.12	1312.1Ø
4	22	5.12	47.85
5	23	10.12	23.7Ø
6	24	17.12	25
7	25	19.12	125.3Ø
8	26	19.12	8ØØ
9	27	19.12	725.85
Z = 1Ø	SB (1Ø,1)	SB (1Ø,2)	SB (1Ø,3)
	SB (1Ø,1) = 28	SB (1Ø,2) = 28,12	SB (1Ø,3) = 33,50

Abb. 24 Neun Schecks des Scheckbuches SB sind bereits ausgefüllt und eingetragen. Der zehnte Scheck befindet sich gerade „in Arbeit"

Mit den Zeilen 2Ø bis 6Ø werden also die Daten eingelesen.
Wir geben die Daten in der vorliegenden Form ein:

Beispiel: 1ØØ DATA 19... 19 ist die Scheck-Nummer!
DATA ... 1.12... ist das Datum 1. Dezember!
DATA ... ... 115 ist der Scheck-Betrag!

Und das ist die vollständige Liste:

```
1ØØ  DATA 19 ,   1.12 ,  115
11Ø  DATA 2Ø ,   2.12 ,  125.3Ø
12Ø  DATA 21 ,   5.12 ,  1312.1Ø
13Ø  DATA 22 ,   5.12 ,    47.85
14Ø  DATA 23 ,  1Ø.12 ,    23.7Ø
15Ø  DATA 24 ,  17.12 ,   25
16Ø  DATA 25 ,  19.12 ,   125.3Ø
17Ø  DATA 26 ,  19.12 ,   8ØØ
18Ø  DATA 27 ,  19.12 ,   725.85
19Ø  DATA 28 ,  28.12 ,    33.5Ø
```

3. Schritt: „Lager" ausräumen

Genauso, wie Sie mit unterschiedlicher Zielsetzung in die Liste der von Ihnen ausgestellten Schecks schauen können – mal interessiert Sie ein ganz bestimmter Scheck, mal vielleicht ein bestimmter Betrag – so können Sie von Ihrem Computer bestimmte Details abrufen. Mit dem entsprechenden Programm, versteht sich!

Lassen Sie sich doch zunächst die ganze Liste noch einmal vorzeigen!

```
200 PRINT"SCHECK-NR","AUSSTELL-DATUM","BETRAG"
220 PRINT" ==========================================
= = = = ="
300 FOR Z = 1 TO 10
310 PRINT SB(Z,1),SB(Z,2),SB(Z,3)
370 NEXT Z
```

Na, ist alles sauber abgebucht?

Mit dem nächsten Programm lassen Sie sich einen bestimmten Scheck heraussuchen. „Suchkriterium" ist dabei die Scheck-Nr.

Verändern Sie Ihr Programm wie folgt:

```
200
220 INPUT"NUMMER DES GEWUENSCHTEN SCHECKS EINGEBEN !"; N
310 IF SB(Z,1) = N THEN 350
320 NEXT Z
350 PRINT SB(Z,1),SB(Z,2),SB(Z,3)
370
```

Es wird für Sie nützlich sein, wenn ich Ihnen den Ablauf dieses Programms im Detail erkläre. In Zeile 300 starten Sie die FOR...-NEXT-Schleife. Der Computer beginnt mit Z = 1, d. h. er durchsucht die erste Zeile nach N.

Dort steht für SB(1,1) die Zahl 19. Angenommen, Sie haben oben für N 21 eingegeben. Der Computer stellt dann in Zeile 310 fest, daß SB(1,1) NICHT 21 ist. Zeile 320 schickt ihn erneut in die Schleife; diesmal mit Z = 2. Wieder fällt der Vergleich in Zeile 310 negativ aus; der Computer setzt das Programm fort mit Z = 3. Und nun findet er für SB(3,1) die Zahl 21. Darauf stellt er in Zeile 310 fest: SB(3,1) IST GLEICH N – nämlich 21 = 21 – und er springt befehlsgemäß zu Zeile 350, die ihm aufträgt, Scheck-Nr, Datum und Betrag zu drucken. Das tut er – aber nur für die Zeile, in der er N = 21 gefunden hat.

Geben Sie doch mal für N einfach 50 ein! Sie wissen, daß diese Zahl NICHT in Ihrer Liste enthalten ist. Der Computer wird das auch herausfinden und ausgeben:

```
?BS ERROR IN 350
```

Das bedeutet sinngemäß: Die Eingabe (für N) liegt außerhalb des DIMensionierten Bereichs!

Sie können ihm auch beibringen, in solchen Fällen eine verständlichere Botschaft auszugeben. Ergänzen Sie das Programm um:

```
330 PRINT"DER SCHECK NUMMER";N;"IST NICHT IN DER LISTE!"
340 END
```

Nun folgt ein Programm, das Schecks, die an einem bestimmten Tag ausgegeben wurden, auf Verlangen ausdruckt.

```
200 INPUT"WELCHES AUSSTELLDATUM ";D
220
310 IF SB(Z,2) = D THEN PRINT SB(Z,1),SB(Z,3)
320 NEXT Z
330
340
350
```

Auch dieses Programm wollen wir uns im Detail ansehen. Sie starten; es erscheint:

WELCHES AUSSTELLDATUM?

Geben Sie ein 19. 12

Dieses Mal durchsucht der Computer *zeilenweise* die Liste, ob er in der *zweiten* Spalte auf 19. 12 trifft. Das geschieht gleich dreimal, nämlich in den Zeilen 7, 8 und 9. Und prompt wird er ausgeben:

```
WELCHES AUSSTELLDATUM? 19. 12
25                    125.3
26                    800
27                    725.85
```

Beachten Sie, daß Sie in Zeile 310 nur verlangt hatten, SB(Z,**1**) und SB(Z,**3**) zu drucken!

Zum Schluß wollen wir das letzte Programm noch ein wenig verzieren:

```
195 PRINT "AUFLISTUNG DER AUSGESTELLTEN SCHECKS."
210 PRINT
220 PRINT
230 PRINT "AM"; D; "WURDEN FOLGENDE SCHECKS AUSGESTELLT:"
240 PRINT "=====================================
= = = = ="
250 PRINT "SCHECK-NR", "BETRAG (DM)"
350 FOR X = 1 TO 1000: NEXT
360 GO TO 195
355 CLS
370 END
```

35.2 Immer noch: Lagertechnik!

Bei Variablen hatten Sie gelernt, daß man durch Anhängen eines Dollar-Zeichens ($) an einen Variablennamen den Computer darauf vorbereitet, daß der Variablen eine Zeichenkette – ein String – zugewiesen werden soll.

Das geht bei indizierten Variablen ganz genauso; am besten, Sie probieren das gleich einmal aus!

```
NEW
 5 CLS
1Ø DIM A$(1Ø)
2Ø FOR X = 1 TO 1Ø
3Ø INPUT A$(X)
4Ø NEXT X
```

Mit diesem Programm bereiten Sie den Computer darauf vor, nacheinander 10 Strings aufzunehmen, für die er Platz im Speicher bereithalten soll.
Für die AUSGABE setzen Sie Ihr Programm fort:

```
45 CLS
5Ø FOR X = 1 TO 1Ø
6Ø PRINT X,A$(X)
7Ø NEXT X
8Ø END
```

IHRE EINGABE		DIE COMPUTER-AUSGABE	
? SOMMER		1	SOMMER
? WINTER		2	WINTER
? HERBST		3	HERBST
?	ENTER	4	
? FRUEHLING		5	FRUEHLING
?	ENTER	6	
?	ENTER	7	
? JA		8	JA
? NEIN		9	NEIN
? VIELLEICHT		1Ø	VIELLEICHT

In der dritten, sechsten und siebten Zeile hatten Sie NICHTS eingegeben, sondern gleich mit ENTER weitergeschaltet. Sie erkennen bei der Ausgabe, daß der Computer genau diese Speicherplätze auch freigehalten hat.

Nun MISCHEN wir in einem Programm einmal numerische und Stringvariable; das sollte Ihnen nicht schwer fallen!

```
10 DIM Z$(5) : DIM N(5)
20 FOR A = 1 TO 5
30 INPUT Z$(A) : INPUT N(A)
40 NEXT A
50 PRINT "MARKE","BESTAND"
60 PRINT" ==================================="
70 FOR A = 1 TO 5
80 PRINT Z$(A),N(A)
90 NEXT A
100 END
```

Wenn Sie dieses Programm starten, wird es 10 Fragen stellen. Und zwar abwechselnd je 5 für Z$(...) und für N(...). Und in dieser Reihenfolge müssen Sie auch antworten!

Also:

```
? ERNTE
? 100
? REVAL
? 0
? WINSTON
? 25
? HB
? 15
? DUNHILL
? 27
```

Und so wird diese „Mini-Liste" ausgegeben:

```
MARKE        BESTAND
= = = = = = = = = = = = =
ERNTE        100
REVAL        0
WINSTON      25
HB           15
DUNHILL      27
```

Nach dieser Einstimmung wollen wir uns auf den folgenden Seiten wieder einmal ein größeres Programm vornehmen, das Sie als Telefonverzeichnis nutzen können. Falls Sie Präsident eines Vereins sind, können Sie – nach entsprechender Abwandlung – auch eine Mitgliederliste daraus bauen. Oder... oder... Ihrer Phantasie sind keine Grenzen gesetzt!

35.3 Der Computer als Telefonverzeichnis

```
 5 CLEAR 1ØØØ
1Ø DIM N $ (5Ø) : DIM A $ (5Ø) : DIM T $ (5Ø)
15 CLS
2Ø PRINT "***** BEDIENUNGSANLEITUNG *****"
25 PRINT
3Ø PRINT "WENN SIE EIN TELEFONVERZEICHNIS ANLEGEN WOLLEN...
.................................DRUECKEN SIE DIE TASTE 1!"
```

In der Zeile 5 CLEAR 1ØØØ schaffen Sie zunächst einmal Platz für Stringvariable. In der Zeile 1Ø DIM... richten Sie darauf für 5Ø Namen usw. Ihr Lager ein. Stellen Sie sich das Zusammenspiel der Zeilen 5 und 1Ø so vor: 5... ist das „*Grundstück*", auf dem Sie mit 1Ø Ihr Lager „bauen".

```
 7Ø INPUT Q
 8Ø ON Q GO TO 1ØØ
1ØØ INPUT "WENN SIE BEREIT SIND; DRUECKEN SIE 'ENTER'!...
..........WENN SIE EINE LISTE BEENDEN WOLLEN, TIPPEN SIE 'ENDE'
ANSTELLE DES NAMENS!"; X
11Ø FOR I = 1 TO 5Ø
115 CLS
12Ø INPUT "GEBEN SIE DEN NAMEN EIN!";N $ (I)
125 IF N $ (I) = "ENDE" THEN A = I : GO TO 16Ø
13Ø INPUT "GEBEN SIE DIE ADRESSE EIN!"; A $ (I)
14Ø INPUT "GEBEN SIE DIE TELEFON-NR. EIN!"; T $ (I)
15Ø NEXT I
16Ø PRINT "DIE LISTE IST ABGESCHLOSSEN"
165 END
```

Die Zeilen 7Ø und 8Ø mögen Ihnen spaßig vorkommen. Aber wir werden das Programm später verlängern!

Ihrer besonderen Aufmerksamkeit empfehle ich die Zeile 125. Sie bewirkt – zusammen mit einer Zeile, auf die ich erst später eingehen kann – daß Sie Ihr Telefonbuch nach jeder beliebigen Anzahl von Namen zwischen 1 und 5Ø – etwa bei 12 – beenden können, ohne daß der Computer sich querlegt.

Mit dem Programmteil bis Zeile 165 können Sie Ihr Telefonbuch anlegen – mit dem folgenden können Sie auch „ins Telefonbuch schauen"!
Wählen Sie für die Eingabe folgendes „Format":

Name	Adresse	Telefon-Nr.
Horst Weber	1 Berlin Steinweg 27	0127-33456

```
165
170 INPUT "WENN SIE DIE BEDIENUNGSANLEITUNG SEHEN WOLLEN:...
.....DRUECKEN SIE 'ENTER'!"; X
180 GO TO 15
 40 PRINT "WENN SIE DAS TELEFONVERZEICHNIS EINSEHEN WOLLEN:...
 .....DRUECKEN SIE DIE TASTE 2"
 80 ON Q GO TO 100, 200
 90 IF Q < 1 THEN 700
 95 IF Q > 2 THEN 700
700 PRINT "SIE HABEN EIN NICHT EXISTIERENDES PROGRAMM AUFGE-
    RUFEN!
.....DRUECKEN SIE 1 ODER 2!"
710 FOR X = 1 TO 1000 : NEXT
720 GO TO 15
```

Sie haben zunächst Zeile 165 aus dem ersten Programmteil wieder gelöscht. Mit 40 vervollständigen Sie Ihre Bedienungsanleitung.
Mit den Zeilen 90, 95 und 700 bis 720 ist wieder die übliche Sperre gegen Falscheingaben eingebaut.

```
200 CLS
210 FOR I = 1 TO A
220 PRINT N $ (I), A $ (I), T $ (I)
230 NEXT I
240 END
```

Und jetzt noch einmal zurück zu Zeile 125! Diese lautete:

```
125 IF N$(I) = "ENDE" THEN A = I : GO TO 160
```

Und jetzt Zeile 210:

```
210 FOR I = 1 TO A
```

Sie hatten – beispielsweise – nach dem 12. Namen das Telefonbuch abschließen wollen. Sie haben deshalb getreu der Bedienungsanleitung für N$(**13**) – für den 13. Namen – ENDE eingegeben. Daraufhin hat der Computer in Zeile 125 A zu I gemacht – also A zu 13! Gleichzeitig ist er in Zeile 160 gesprungen und hat ausgegeben: DIE LISTE IST ABGESCHLOSSEN.

Wenn Sie jetzt im folgenden dritten Programmteil einen Namen suchen, dann wird der Computer das *ganze* Programm nach ihm durchsuchen. Ihr Programm – das ja für 50 Namen etc. DIMensioniert war – ist aber durch Ihre obige Eingabe auf 13 (!) Plätze geschrumpft. Und um unnötige Sucharbeit zu sparen, heißt es jetzt in Zeile 210: FOR I = 1 TO A. Also: für I von 1 bis 13! Bei unserem Programm sparen wir durch diesen Trick weder Strom noch Zeit. Bei umfangreichen Programmen kann zumindest die Such*zeit* ins Gewicht fallen!

Nachdem Sie nun in der Lage sind, das ganze Verzeichnis einzusehen, wollen wir das Programm im nächsten Abschnitt so erweitern, daß Sie auch einen ganz bestimmten Namen aussuchen können.

```
240
250 INPUT "WENN SIE DIE BEDIENUNGSANLEITUNG SEHEN WOLLEN:...
.............................................DRUECKEN SIE 'ENTER'!";´X
260 GO TO 15
 50 PRINT "WENN SIE EINEN BESTIMMTEN NAMEN SUCHEN:...
 ...............................DRUECKEN SIE DIE TASTE 3"
 80 ON Q GO TO 100, 200, 300
 95 IF Q > 3 THEN 700
700 PRINT "SIE HABEN EIN NICHT EXISTIERENDES PROGRAMM AUFGE-
    RUFEN!...
.......................................DRUECKEN SIE 1, 2 ODER 3!"
300 CLS
310 INPUT "GEBEN SIE DEN NAMEN EIN!";Z $
320 FOR I = 1 TO A : IF N $ (I) = Z $ THEN 350
330 NEXT I
340 PRINT "DER NAME STEHT NICHT IM VERZEICHNIS!": GO TO 360
350 PRINT N $ (I), A $ (I), T $ (I)
360 PRINT: PRINT "WOLLEN SIE EINEN ANDEREN NAMEN?...
....................DRUECKEN SIE 1, ANDERNFALLS 0!" INPUT X
370 IF X = 1 THEN 300 ELSE 15
```

In Zeile 310 werden Sie aufgefordert, denjenigen Namen einzugeben, für den Sie Adresse und Telefon-Nummer sehen wollen.

Hierzu gibt es eine wichtige Regel, die für ALLE Vergleichsoperationen gilt. Sie müssen den Namen so ABRUFEN, wie Sie ihn im ersten Programmteil EINGEGEBEN hatten.

Hatten Sie beispielsweise HORST WEBER eingegeben, und Sie fragen jetzt nach WEBER, HORST oder schlicht nach WEBER, dann wird der Vergleich in Zeile 320 negativ ausfallen, der Computer wird feststellen, daß die beiden Strings NICHT identisch sind und ausgeben: DER NAME STEHT NICHT IM VERZEICHNIS.

Den in Zeile 310 eingegebenen Namen setzen Sie = Z$. In Zeile 320 sucht der Computer in dem von Ihnen mit Namen belegten Speicherbereich, ob er ein N$(I) findet, das gleich Z$ ist. Vergleichen Sie hierzu auch meine Aussagen zu Zeilen 125 und 210 des vorigen Programmteils!

Die Zeile 370 lautet:

```
370 IF X = 1 THEN 300 ELSE 15
```

„ELSE" steht hier nicht für einen weiblichen Vornamen, sondern für „anderen-

falls...". Übersetzt heißt die Zeile: „Wenn X=1, dann gehe zu Zeile 300, anderenfalls zu Zeile 15". Falls Ihr Computer diesen Komfort nicht bietet, können Sie auch sagen:

```
370 IF X = 1 THEN 300
380 IF X<>1 THEN 15
```

Im letzten Teil unseres Programms schaffen Sie sich die Möglichkeit, eine Eintragung auch zu ändern: schließlich kann Ihr Freund ja mal umziehen!

```
 60 PRINT "WENN SIE AENDERUNGEN MACHEN WOLLEN:...
.........................DRUECKEN SIE DIE TASTE 4"
 80 ON Q GO TO 100, 200, 300, 400
 95 IF Q > 4 THEN 700
700 PRINT "SIE HABEN EIN NICHT EXISTIERENDES PROGRAMM AUFGE-
    RUFEN!...
.......................................DRUECKEN SIE 1, 2, 3 ODER 4!"
400 CLS
410 PRINT "GEBEN SIE DEN ZU AENDERNDEN NAMEN EIN!"
420 INPUT Z $
430 FOR I = 1 TO A: IF Z $ = N $ (I) GO TO 460
440 NEXT I
450 PRINT "DER NAME IST NICHT VORHANDEN!" : GO TO 510
460 PRINT "GEBEN SIE AUF '?' DIE NEUEN DATEN EIN!...
    ...NAME, ADRESSE, TEL-NR! KEINE KOMMAS!"
470 INPUT N $ (I), A $ (I), T $
480 PRINT "DIE NEUE EINTRAGUNG LAUTET:"
490 PRINT
500 PRINT N $ (I), A $ (I), T $ (I)
510 INPUT "ZU EINER WEITEREN KORREKTUR DRUECKEN SIE 1!...
...............................FOLGT KEINE WEITERE KORREKTUR; DRUECKEN SIE 0!"; X
520 IF X = 1 GO TO 400
530 GO TO 15
```

35.4 Der Computer als Lagerverwalter

Da Sie inzwischen schon ein anspruchsvoller Programmierer sind, werden Sie möglicherweise zwei Eigenschaften der letzten Programme gestört haben:

- Die als indizierte Variable eingegebenen „Daten" waren verschwunden, sobald etwa durch einen Eingabefehler das Programm „ausgestiegen" war oder Sie es per LIST betrachten wollten.
- Sie mußten im voraus einen Speicherbereich genau DIMensionieren.

Abhilfe für den ersten Fall werden wir beim nächsten Programm dadurch schaffen, daß wir die Daten mit READ...DATA... als LISTE von Daten eingeben.

Was das DIMensionieren angeht, wollen wir dem – als „Vorübung" – gleich einmal abhelfen. Geben Sie ein:

```
NEW
10 INPUT N
20 DIM A(N)
```

Jetzt DIMensionieren Sie also A(1), A(2), A(3)...A(N) – wobei die ANZAHL der indizierten Variablen von N abhängt; N aber geben Sie in Zeile 10 nach Ihrer Wahl – oder besser, nach Ihren jeweiligen Bedürfnissen – ein.

Damit der Computer mit uns „reden" kann, ändern Sie:

```
10 INPUT"WIEVIEL PLATZ SOLL ICH RESERVIEREN";N
```

Das Programm geht weiter mit dem Eingabeteil:

```
30 FOR X = 1 TO N
40 INPUT A(X)
50 NEXT X
```

Und jetzt folgt der Ausgabeteil:

```
55 PRINT "X=,,,"A="
60 FOR X = 1 TO N
70 PRINT X,,,A(X)
80 NEXT X
90 END
```

Machen Sie einmal einen Probelauf; damit es nicht langweilig wird, beantworten Sie die Frage nach N mit 5.

Ihre Eingabe:	...die Ausgabe:	
? 10	X=	A=
? 20	1	10
? 30	2	20
? 40	3	30
? 50	4	40
	5	50

Daten verarbeiten heißt ja wohl unter anderem, daß der Computer eingegebene Daten „verarbeiten", mit ihnen etwas tun kann. Bis jetzt hat er sie meist nur hingeschrieben, von „Verarbeiten" kann noch kaum die Rede sein.

Wir wollen dem abhelfen und uns einmal die „Summe aller A" ausrechnen lassen. Ergänzen Sie das letzte Programm wie folgt:

```
52 S = 0
55 PRINT"X=","A=","S="
65 S = S + A(X)
70 PRINT X,A(X),S
```

Geben Sie die gleichen Werte wie oben ein! (Also N = 5; A(X) von 10 bis 50)
Sie werden folgendes Ergebnis erhalten:

X=	A=	S=
1	1Ø	1Ø
2	2Ø	3Ø
3	3Ø	6Ø
4	4Ø	1ØØ
5	5Ø	15Ø

Sie werden leicht im Kopf nachvollziehen können, daß der Computer in der letzten Spalte immer zu dem „aktuellen" A die Summe der A aus den vorangegangenen Zeilen hinzuaddiert hat. „Schuld" daran ist unsere bekannte Schleife, in der zunächst S (für Summe) = Ø gesetzt und bei jedem weiteren Durchgang S um A(...) erhöht wurde. Dieses trickreiche Spiel sollten Sie sich merken!

Und nun zu unserem Programm, das den Computer als Lagerverwalter zeigt.

Und zwar soll er in einer Schraubengroßhandlung eine Liste für alle angebotenen metrischen Sechskantschrauben führen. Sie wünschen sich die Ausgabe in folgendem „Format":

Lager.-Nr.	Abmessungen	Lagerbestand	Einzelpreis

Damit Sie nach Herzenslust in dieser Liste „blättern" können, hatte ich die Verwendung einer DATEN-LISTE in Aussicht gestellt. Es ist immer gut, wenn man auf bereits erworbenes Wissen zurückgreifen kann. Wir tun das hier, indem wir aus unserem „Scheckbuch-Programm" den Eingabeteil und die Datenliste übernehmen. Unser Ehrgeiz läßt natürlich wörtliches „Abschreiben" nicht zu!

```
NEW
1Ø PRINT"                    LAGERLISTE 1"
2Ø PRINT"      FUER METRISCHE SECHSKANTSCHRAUBEN"
3Ø PRINT" ==================================
= = = = "
4Ø PRINT
5Ø PRINT
6Ø INPUT"WIEVIELE POSITIONEN WERDEN GELAGERT";N
7Ø DIM SL$(N,4)
```

Sie meinen, das wäre zuviel der Variation und Sie verstehen das nicht? Ich erklär's Ihnen!

Die Zeile 6Ø werden Sie sicher als die Frage des Computers erkennen, mit der er nach dem zu DIMensionierenden Platz fragt. Nun schauen Sie oben das von Ihnen gewünschte „Format" an! Ganz offensichtlich brauchen Sie für jede Schrauben-Position EINE ZEILE. Also gilt die Frage der Zeilenzahl; Z(1...)...Z(N).

Jede Zeile hat aber 4 Spalten; deshalb ist S (für Spalte) immer 4 – egal, wie Sie N DIMensionieren! Somit heißt Zeile 70 im Klartext: „DIMensioniere (für) eine Schraubenliste mit N Zeilen und 4 Spalten..."
Können wir weitermachen?

```
80 FOR Z = 1 TO N
90 FOR S = 1 TO 4
100 READ SL$(Z,S)
110 NEXT S
120 NEXT Z
```

Nun können Sie daran gehen, Ihre Datenliste einzugeben. Da eine Sechskantschraube mit 10 mm Durchmesser und 10 mm Länge unter Technikern eine Schraube M 10 x 10 ist, geben wir die Daten als Strings ein – deshalb SL$! Sie können natürlich die Liste beliebig lang machen; aber ich glaube, für Übungszwecke reichen 5 Positionen aus!

```
400 DATA 200,M 10 x 15, 1000, 10
410 DATA 201,M 10 x 20, 2000, 20
420 DATA 202,M 20 x 10, 1000, 30
430 DATA 203,M 20 x 50, 1500, 50
440 DATA 204,M 20 x 100, 100, 20
```

Noch etwas: regen Sie sich bitte nicht über die Preise auf! Das sind schlicht und einfach glatte Zahlen, die es Ihnen ermöglichen sollen, im Zweifel eine Kopfrechnung zu machen! Und damit zur „Bearbeitung"! Zunächst wollen wir die Liste im ganzen sehen! Der Computer zeigt sie ihnen, wenn Sie folgendes Programm starten:

```
150 Bitte Text von Zeile 10 eingeben! Dann weiter:
160 ...wie Zeile 20
170 ...wie Zeile 30
160 PRINT
170 PRINT
180 PRINT"LAGER.-NR","ABMESSUNG","BESTAND","EINZELPREIS"
190 PRINT"= = = =
```

Zeile 21Ø: Der Computer schreibt: SL$(1,1),SL$(1,2),SL$(1,3),SL$(1,4),
das heißt, von der ersten Zeile alle 4 Spalten.
Zeile 22Ø: heißt: zurück zu Zeile 2ØØ

2. Durchgang:
Zeile 2ØØ: Z ist 2
Zeile 21Ø: Der Computer schreibt SL$(2,1),SL$(2,2),SL$(2,3),SL$(2,4)
Und so weiter...

N-ter Durchgang:
Zeile 2ØØ: Z ist N; das war aber von Ihnen in Zeile 6Ø auf 5 festgelegt!
Zeile 21Ø: Der Computer schreibt SL$(5,1)...usw. usw.

Nun wollen Sie die Schrauben ja verkaufen – das heißt, der Bestand verändert sich laufend. Sie suchen also eine Möglichkeit, Ihre DATA-Liste bequem zu ändern. Das ist beim Arbeiten mit READ...DATA... ganz einfach! Schreiben Sie:

LIST 4ØØ–
...und schon haben Sie Ihre Datenliste vor sich! Wollen Sie in der letzten Zeile etwas ändern, so schreiben Sie:

44Ø DATA... und nun den *ganzen* Text; d. h., was NICHT geändert werden soll, überschreiben Sie mit dem alten, was geändert werden soll, mit dem neuen Text!

Eine „Inventur" zu machen – das fürchten heute noch viele Kaufleute. Da muß man nämlich die Schrauben *zählen* und (nach Menge mal Einzelpreis) *bewerten*. Für Sie als Computerbesitzer ist das kein Problem!

Ändern Sie Ihr Programm wie folgt ab:

```
18Ø PRINT "LAGER.-NR", "ABMESSUNG","BESTAND","WERT"
21Ø PRINT SL$(Z,1),SL$(Z,2),SL$(Z,3),SL$(Z,3)*SL$(Z,4)
```

Sie bemerken, daß nur die *vierte Spalte* geändert wurde? Unglücklicherweise hat der Computer ausgegeben:

LAGER.-NR.	ABMESSUNG	BESTAND	WERT
2ØØ	M 1Ø x 15	1ØØØ	

? TM ERROR IN 21Ø

„TM ERROR" hatten wir noch nicht. Das Handbuch hat mich darüber aufgeklärt, daß ich einen ganz üblen Schnitzer gemacht habe. „TM" steht für „TYPE MISMATCH"... und das heißt sinngemäß „...man behandelt STRINGS nicht wie „NUMERISCHE VARIABLE" – indem man sie miteinander multipliziert. Das ist ja auch ganz logisch; multiplizieren Sie doch mal „MITTWOCH" mit „DONNERSTAG"...

Nun habe ich zunächst ein wenig geschwitzt. Ich möchte Ihnen das ersparen, und lieber etwas Neues einführen!

Machen Sie (altes Programm stehenlassen!!) ein paar Fingerübungen:

```
1000 A$ = "1000"
1010 B$ = "WERT 1000"
1020 PRINT A$,B$,VAL (A$), VAL (B$)
```

Dann

RUN 1000

Das Ergebnis:

```
1000                WERT 1000          1000                0
```

Zur Erklärung: VAL(...) steht für VALue – und das heißt WERT.

Nach dem Befehl VAL(...) muß in (...) ein STRING stehen. 1000 ist KEIN STRING. Aber "1000" ist einer; schließlich haben Sie ja extra "....." verwendet, um aus 1000 den String "1000" zu machen!

Aber "WERT 1000" ist doch ein STRING – warum kommt dann 0 raus?...

Nein, so begreift das kein Mensch! Ich versuche am besten, Ihnen einmal in verständliches Deutsch zu übersetzen, was ich zu VAL(...) im Handbuch gefunden habe.

- Bei VAL(...) muß in (...) ein String stehen.
- Dieser muß jedoch – als Zahlenwert betrachtet – einen SINN ergeben.

Jetzt zu unserem Zwischenprogramm:

1000 A$="1000" ist ein String. 1000 für sich betrachtet, macht auch Sinn – nämlich die Zahl 1000

Deshalb ergibt:

1020 PRINT...VAL(A$) den Ausdruck 1000

1010 B$="WERT 1000" ist eindeutig auch ein String. Allerdings macht WERT 1000 als ZAHL KEINEN SINN! Und deshalb ergibt

1020 PRINT...VAL(B$) schlicht: „NICHTS" (0)

Vielleicht werden die letzten Klarheiten beseitigt, wenn Sie dieses „Wissen" in das mißlungene Inventurprogramm einbauen!

```
210 PRINT SL$(Z,1),SL$(Z,2),SL$(Z,3),VAL(SL$(Z,3))*VAL(SL$(Z,4))
```

Jetzt wird's wohl klarer!

Nach der Datenliste, Zeile 400 hat z. B. SL$(1,3) den ZAHLENWERT 1000. Und VAL(SL$(1,3)) ist dann der „VALue", der „Zahlenwert" des Strings SL$(1,3). Mit VAL(...) können Sie einen als String „versteckten" Zahlenwert zum Vorschein bringen.

Dasselbe passiert mit SL$(1,4); durch VAL(SL$(1,4)) wird daraus 100. Und nun können wir beide multiplizieren! Zuerst aber noch

```
230 END
```

Rechnen Sie die Zahlen in der 4. Spalte nach! Es handelt sich tatsächlich um die Produkte aus Menge und Einzelpreis!

Nun können Sie einen Taschenrechner nehmen, und die Werte der 4. Spalte aufaddieren – Sie haben dann den Gesamtwert des Schraubenlagers.
Eleganter geht es natürlich mit einem entsprechenden Computerprogramm. Mit unserem Wissen um VAL(...) wollen wir es wagen!

```
230 RESTORE
```

Da wir mit der *alten* Datenliste etwas *Neues* anfangen wollen, ist RESTORE notwendig.

```
240 S = 0
```

S steht für Summe. Beim Beginn des Programms ist diese noch 0.

```
250 Z = 1
```

Z soll jetzt die Zeilen zählen.

```
260 READ SL$(R,1),SL$(R,2),SL$(R,3),SL$(R,4)
270 S = S + (VAL(SL$(Z,3))*VAL(SL$(Z,4)))
280 Z = Z + 1
290 IF Z <=N THEN 270
300 PRINT
310 PRINT"DER GESAMT-LAGERWERT IST ";S;" DM"
320 PRINT" ==========================================
=====
330 END
```

Geschafft! Auch dieses Programm erkläre ich Ihnen gern noch im Klartext. Da Ihre Datenliste in Viererblocks aufgestellt ist, müssen auch alle vier Elemente gelesen werden, obwohl Sie nur die letzten beiden für die Weiterverarbeitung benötigen.

Der Computer beginnt mit den Werten 0 für S und 1 für Z. Damit rechnet er in Zeile 270 S = 0 + (1000 mal 10).

In Zeile 280 wird Z um 1 erhöht; in Zeile 290 geht es zurück zu Zeile 270. Dort wird jetzt gerechnet: S = 10000 + (2000 mal 20). (Vergleichen Sie die Werte in der Datenliste!) das geht solange, wie Z kleiner oder gleich N ist. Erst der letzte Wert für S – das ist dann der Gesamtwert – wird in Zeile 310 hingeschrieben.

Was haben Sie dazugelernt?

- Sie haben erfahren, was man unter „indizierten Variablen" versteht und wie man damit umgeht.
- Alle indizierten Variablen mit dem gleichen Namen – wie A(1), A(2), A(3)...A(N) nennt man ein „Feld", einen „Bereich" oder (englisch) „Array".
- Sie kennen den Unterschied zwischen:
  - *eindimensionalen Feldern.* A(I). Im Beispiel ab Seite 150 war das Lager aus einzeiligen Fächern auf dem Boden aufgebaut.

– *zweidimensionalen* Feldern. A(N,M). Im Beispiel ab Seite 156 war das unser Lager, das auch in die Höhe geht.

- Wenn Sie mit indizierten Variablen arbeiten, müssen Sie dafür sorgen, daß der Computer dafür Speicherplatz bereithält. Sie beauftragen ihn dazu mit:
  DIM A(N) – wobei N von Ihnen vorgegeben wird.
  DIM A(17) – hier wollen Sie 17 Plätze reservieren.
  DIM S(10,4) – hier dimensionieren Sie ein Feld, bestehend aus 10 Reihen mit je 4 Spalten.
  DIM... ist gewissermaßen ein Geschäft mit Ihrem Computer auf Gegenseitigkeit.
  – *Sie* halten sich an den selbst gesetzten Rahmen...
  – *Er* garantiert dafür den bestellten Speicherplatz.
- Sie können numerische Variable A, A3, Z... usw. indizieren zu A(I), A3(N), Z(A)...
  aber ebenso
- String-Variable wie A$(I), A1$(N) oder Z$(A).
- Indizierte Variable können Sie auch „bearbeiten". In Beispielen hatten Sie eine Liste von Variablen addiert oder die Zahlenwerte von Stringvariablen miteinander multipliziert.
- Dabei haben Sie mit VAL(...) einen Befehl mit einer interessanten Eigenschaft kennengelernt:
- hinter VAL(...) muß in (...) ein String folgen, der – für sich betrachtet – als Zahl einen Sinn ergibt.
- Das ist bei VAL(SL$(1,4)) der Fall, wenn SL$(1,4) den Wert 1000 hat. Also als Zahl einen Sinn macht!
- Das ist bei VAL(B$) NICHT der Fall, wenn B$ "WERT 250" ist. Weil WERT 250 keine Zahl ist!

36 Die Logik der Steuerungstechnik

Sie lernen wieder etwas hinzu!
Ich habe Ihnen ein Schaltbild aufgezeichnet (*Abb. 25*). Auch wenn Sie kein Elektriker sind, werden Sie erkennen, daß die Lampe nur leuchten kann, wenn *alle drei* Schalter S1, S2 und S3 geschlossen sind.

In der Elektrotechnik gibt es eine Disziplin, die sich mit digitaler Steuerungstechnik befaßt. Ein Fachmann dieser Richtung würde das Schaltbild mit folgenden Worten beschreiben:
„Wenn S1 UND S2 UND S3 geschlossen sind, dann leuchtet die Lampe."
Das Schaltbild symbolisiert eine sogenannte UND-Schaltung – wobei Sie wissen müssen, daß man in elektrischen/elektronischen Schaltbildern die Schaltung im Ruhezustand, mit unbetätigten/stromlosen Schaltelementen zeigt.

Für Sie als angehenden Computerfachmann ist es wichtig zu lernen, daß dieses UND unter gar keinen Umständen im Sinne von PLUS verstanden werden darf.

Und nun zu Ihrem Computer. Der versteht UND nämlich auch. Sie überprüfen das am besten auf der Grundlage des folgenden Programms:

```
NEW
5 CLS
10 PRINT"SCHALTER EIN = 1 , SCHALTER AUS = 0"
15 PRINT
20 INPUT"IST SCHALTER S1 GESCHLOSSEN";A
30 INPUT"IST SCHALTER S2 GESCHLOSSEN";B
40 INPUT"IST SCHALTER S3 GESCHLOSSEN";C
```

Und nun lernen Sie die neue Fähigkeit des Computers kennen:
Da der Computer nur Englisch versteht, will er auch den Begriff UND übersetzt haben. Und das heißt AND.

```
50 IF A = 1 AND B = 1 AND C = 1 THEN 80
60 PRINT"DIE LAMPE LEUCHTET NICHT"
70 END
80 PRINT"DIE LAMPE LEUCHTET"
90 END
```

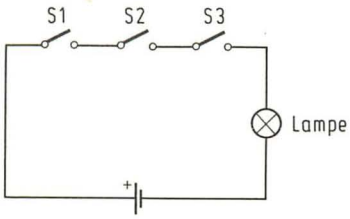


Abb. 25 Eine UND-Schaltung. Die Lampe leuchtet nur, wenn Schalter 1 UND Schalter 2 UND Schalter 3 geschlossen sind

Wenn Sie dieses Programm testen, werden Sie erkennen, daß der Computer nur dann meldet:
DIE LAMPE LEUCHTET
Wenn Sie für A und B und C jeweils 1 für „Schalter geschlossen" eingegeben hatten.

Wenn Sie das Schaltbild kritisieren, weil drei Schalter in Reihe Geldverschwendung wären, dann kann ich Ihnen auch ein anspruchsvolleres zumuten (*Abb. 26*).

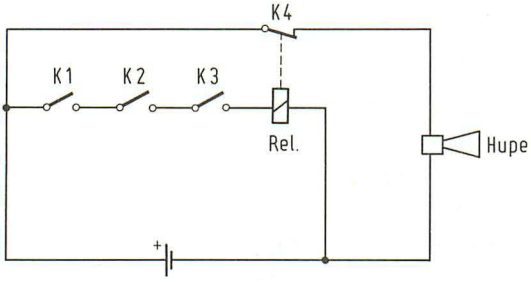


Abb. 26 Eine einfache Alarmanlage; Wenn die Schalter K1, K2, K3 alle geschlossen sind, bekommt das Relais Rel. Strom, der Ruhekontakt K4 ist geöffnet und die Hupe bleibt still. Wehe aber, einer der Schalter K1 bis K3 wird geöffnet

Das könnte etwa das Schaltbild einer Alarmanlage sein. Wenn alle drei (Fenster-)Kontakte K1, K2 und K3 geschlossen sind, dann ist das Relais Rel angezogen, und dessen Kontakt K4 *geöffnet* – was bewirkt, daß die Hupe stumm bleibt.

Wenn einer der Kontakte K1, K2, K3 offen ist, wird das Relais Rel stromlos; der RUHEkontakt K4 (der heißt so, weil er im RUHEzustand, also bei stromlosem Relais Rel *geschlossen* ist) schließt sich... und wenn Sie nicht zu Hause sind, werden Ihre Nachbarn wach und verscheuchen den Einbrecher. Bauen Sie das vorige Programm um!

```
20 INPUT"IST K1 GESCHLOSSEN";A
30 INPUT"IST K2 GESCHLOSSEN";B
40 INPUT"IST K3 GESCHLOSSEN";C
60 PRINT"ALARM! ES IST JEMAND IM HAUS!"
80 PRINT"ALLES IN ORDNUNG! GUTE NACHT!"
```

Es ist natürlich wenig geistreich, durch das Haus zu laufen...
...zu prüfen, ob alle Fenster geschlossen sind...
...dies dem Computer mitzuteilen...
...um von ihm zu erfahren, daß alle Fenster geschlossen sind. Für Sie als Programmierer kommt es darauf an, zu lernen, daß der Computer zu dieser Art *logischer Entscheidungen* fähig ist.

Tatsächlich könnten Sie mit einigem Aufwand an „Hardware" den Computer zum Hirn einer Alarmanlage machen.

- Sie müßten dazu über ein spezielles „Interface" den Computer „an die Fenster und Türen anschließen können"...
- Sie müßten den Computer den Schaltzustand der entsprechend präparierten Kontakte „abfragen" lassen...
- Der Computer würde dann ggf. mächtig Alarm schlagen (lassen).

Nun zu einem anderen Schaltbild (*Abb. 27*). Hier sind die drei Schalter S1, S2 und S3 parallel geschaltet. Die Lampe leuchtet auf, wenn S1 ODER S2 ODER S3 geschlossen sind. Auch das kann der Computer unterscheiden. Nachdem Sie ODER zu OR übersetzt haben, können Sie folgendes Programm schreiben:

```
NEW
5 CLS
10 PRINT"SCHALTER EIN = 1 ; SCHALTER AUS = 0"
20 INPUT"ZUSTAND SCHALTER S1";A
30 INPUT"ZUSTAND SCHALTER S2";B
40 INPUT"ZUSTAND SCHALTER S3";C
```

Und nun die „Entscheidung"!

```
50 IF A = 1 OR B = 1 OR C = 1 THEN 80
60 PRINT"DIE LAMPE IST DUNKEL"
70 END
80 PRINT"DIE LAMPE LEUCHTET"
90 END
```

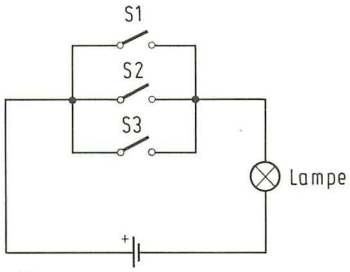


Abb. 27 Eine ODER-Schaltung: Die Lampe leuchtet auf, sobald S1 ODER S2 ODER S3 geschlossen ist

Jetzt gibt der Computer aus:
DIE LAMPE LEUCHTET
wenn Sie für A ODER für B ODER für C eine 1 eingeben.
Nur wenn Sie A=0, B=0, C=0 eingeben, kommt die Antwort:
DIE LAMPE IST DUNKEL
Weil dann die Bedingung in Zeile 50 NICHT erfüllt ist.
Sicherlich können Sie sich vorstellen, daß die erwähnten Steuerungstechniker in der Regel mit Anlagen zu tun haben, die mehr als drei Kontakte haben. Manchmal – in großen Steuerungen – können es hunderte sein. Wollten die Steuerungstechniker ihre Schaltungen dann verbal beschreiben – wie in unseren Beispielen – dann kämen völlig unverständliche Romane heraus.

Sie haben deshalb für AND und OR Kurzzeichen; sie sagen: (Bitte festhalten!)

für AND ——————————————————————→ *
für OR ——————————————————————→ +

Fragen Sie nicht, was daran „logisch" ist. Stellen Sie sich einfach vor, sie verwenden * für AND, damit SIE das nicht mit PLUS verwechseln. Immerhin versteht der Computer auch diese Kurzzeichen!

Ersetzen Sie in Ihrem letzten Programm (zu Abb. 27) Zeile 50 durch:

```
50 IF (A=1)+(B=1)+(C=1) THEN 80
```

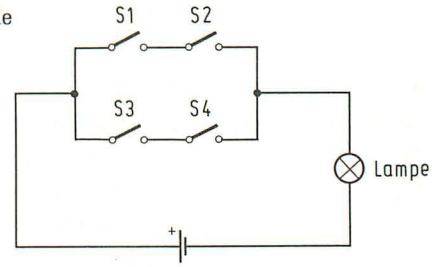


Abb. 28 Wann leuchtet hier die Lampe?

und im ersten Beispiel (zu Abb. 25):
50 IF (A=1)*(B=1)*(C=1) THEN 80

Achten Sie in beiden Fällen auf die Klammern! Und versuchen Sie, wie das arbeitet.

Mein Computer reagiert auf Langschrift und auf Kurzschrift. Wenn Ihr Gerät das auch tut, können Sie zufrieden sein. Wenn nicht – wird es bestimmt auf die eine oder die andere Art der Eingabe reagieren.

Damit Sie etwas zu knobeln haben, noch ein Schaltbild (*Abb. 28*)!
Preisfrage: Wann leuchtet L? Schauen Sie dazu auf die folgende „Wahrheitstabelle" – in der 1 wieder für EIN, 0 wieder für AUS steht (*Abb. 29*).

Der Tabelle können Sie entnehmen, daß es für die vier Schalter 4 x 4 = 16 Zustandskombinationen gibt – aber nur in sieben Fällen leuchtet die Lampe auf.

Wenn Sie die *Bedingung* stellen, daß immer *zwei* Schalter eingeschaltet sein müssen, verbleiben nur zwei Möglichkeiten (Zeile 4 und 13 der Abb. 29).
In Worten: wenn (S1 UND S2) ODER (S3 UND S4)
eingeschaltet sind, leuchtet die Lampe. Für den Computer übersetzt, ergibt das:

```
100 IF A=1 AND B=1 OR C=1 AND D=1 THEN 200
180 PRINT"DIE LAMPE IST DUNKEL"
190 END
200 PRINT"DIE LAMPE LEUCHTET"
```

Und nun die Schalterabfrage:

```
5 CLS
10 PRINT"EIN = 1 ; AUS = 0"
15 PRINT
20 INPUT"ZUSTAND S1";A
30 INPUT"ZUSTAND S2";B
40 INPUT"ZUSTAND S3";C
50 INPUT"ZUSTAND S4";D
```

Prüfen Sie nach – mit einigen Durchläufen – ob Zeile 100 das richtige Ergebnis liefert!

Nr.	S1	S2	S3	S4	Lampe
1	0	0	0	0	0
2	0	0	0	1	0
3	0	0	1	0	0
4	0	0	1	1	1
5	0	1	0	0	0
6	0	1	0	1	0
7	0	1	1	0	0
8	0	1	1	1	1
9	1	0	0	0	0
10	1	0	0	1	0
11	1	0	1	0	0
12	1	0	1	1	1
13	1	1	0	0	1
14	1	1	0	1	1
15	1	1	1	0	1
16	1	1	1	1	1

Abb. 29 Die Wahrheitstabelle zum letzten Bild. Geschlossen bedeutet 1, offen bedeutet 0

Für ein weiteres Beispiel wollen wir zulassen, daß jeder Schalter beliebig betätigt werden darf. Dann gibt es nach der Tabelle sieben Kombinationen, bei denen die Lampe leuchtet. Machen Sie dafür ein Programm:

```
100 IF C=1 AND D=1 AND B=1 THEN 200
110 IF B=1 AND C=1 AND D=1 THEN 200
120 IF A=1 AND C=1 AND D=1 THEN 200
130 IF A=1 AND B=1 THEN 200
140 IF A=1 AND B=1 AND D=1 THEN 200
150 IF A=1 AND B=1 AND C=1 THEN 200
160 IF A=1 AND B=1 AND C=1 AND D=1 THEN 300
210 END
300 PRINT"ALLE VIER SCHALTER SIND GESCHLOSSEN"
310 END
```

Wenn Sie dieses Programm genügend oft durchgespielt haben, werden Sie an seiner Qualität zweifeln.

Sie sind auf dem besten Wege zum Programmierer! Es tut mir leid, daß ich Sie aus pädagogischen Gründen auf eine falsche Fährte geleitet habe. Aber wenn Sie sich schon für das Programm auf die sieben Möglichkeiten, bei denen die Lampe leuchten kann, beschränken, müssen Sie die möglichen Zustände *aller vier Schalter* im Programm berücksichtigen.

Wollen Sie die sechs Zeilen noch mal ändern?

```
100 IF C=1 AND D=1 AND A=0 AND B=0 THEN 200
110 IF B=1 AND C=1 AND D=1 AND A=0 THEN 200
120 IF A=1 AND C=1 AND D=1 AND B=0 THEN 200
130 IF A=1 AND B=1 AND C=0 AND D=0 THEN 200
140 IF A=1 AND B=1 AND C=0 AND D=1 THEN 200
150 IF A=1 AND B=1 AND C=1 AND D=0 THEN 200
```

Und damit Sie alles in Ruhe überprüfen können, noch:

```
190 GO TO 310
210 GO TO 310
310 FOR X = 1 TO 1000 : NEXT
320 GO TO 5
```

Was haben Sie dazugelernt?

- Wie Sie mit Hilfe der Begriffe AND, OR *logische Operationen* durchführen können.
  In Programmen gibt es dafür zwei verschiedene "Schreibweisen";

- Langschrift
  IF A=1 AND B=1 THEN...
  IF A=1 OR B=1 THEN...
- Kurzschrift
  IF (A=1)*(B=1) THEN...
  IF (A=1)+(B=1) THEN...

Also: Für AND gilt das Zeichen *
Für OR gilt das Zeichen +

Wichtig

- AND nicht mit UND im Sinne von PLUS verwechseln!
- Bei „Kurzschrift" die Klammern nicht vergessen!

37 Der Computer als Vermögensberater

Ich habe eine Tochter. Sie heißt Susanne. Ihre Freunde – und alle, die sie mögen – nennen sie Susi. Als Vorbereitung auf ihr Studium lernt Susi zur Zeit Bankkaufmann. Wenn sie Schalterdienst hat, kann es – beispielsweise – geschehen, daß eine ältere Dame an ihren Schalter kommt und fragt: „Fräulein Susi, ich habe einen Enkel bekommen. Von meiner Rente kann ich DM 50.– im Monat abzweigen. Wenn ich die bei Ihnen anlege: Wieviel Geld bekommt mein Enkel, wenn er 18 Jahre alt ist?" Susi wird dann ein wenig nervös, blättert in Tabellen, bemüht auch ihren Tischrechner. So, wie ich Susi kenne, erhält die alte Dame aber nach einer Weile eine zufriedenstellende Auskunft. Wieviel leichter hätte sie es, wenn sie – wie Sie – einen Computer hätte!

Für solche und ähnliche Fragen sei ein Computerprogramm für Susi geschrieben! Die Grundlagen dazu habe ich in Susis Fachkundebuch gefunden. Darin werden vier Eingangsgrößen bearbeitet; mit drei bekannten läßt sich jeweils die vierte Größe nach den Formeln, die ich Ihnen gleich aufschreibe, errechnen. Die vier Größen sind:

1. *Das Anfangskapital KA:* Das ist der Geldbetrag, den Sie für eine bestimmte Zeit anlegen wollen. Man könnte auch schlicht sagen: KA ist der Betrag, den Sie sparen wollen.

2. *Die Laufzeit N:* N ist die Anzahl der Jahre, für die Sie den Betrag KA der Bank anvertrauen wollen. Die arbeitet dann mit Ihrem Geld – als Beteiligung erhalten Sie dafür von der Bank Zinsen.

3. *Zinssatz Z:* Dieser sagt aus, wieviel Prozent Zinsen Ihnen die Bank im Jahr zusichert.

4. *Das Endkapital KE:* Das ist der Betrag, den Sie nach der Laufzeit N von der Bank zurückerhalten. In KE steckt natürlich Ihr Anfangskapital KA, vermehrt um die Zinsen und Zinseszinsen über die Laufzeit N.

Wie erwähnt: Sind drei dieser Größen bekannt, läßt sich die vierte errechnen.

Und nun zu den Formeln, die ich bereits computergerecht aufbereitet habe und die Sie – bitte – kommentarlos entgegennehmen wollen!

Endkapital KE

KE = KA * (1 + Z/1ØØ)↑N

Laufzeit N
N = (LOG(KE) – LOG (KA))/ LOG(1 + Z/1ØØ)
Zinssatz Z
Z = 1ØØ * ((KE/KA) ↑ (1/N)–1)
Anfangskapital KA
KA = KE/((1+Z/1ØØ)↑ N)
Damit ist eigentlich das Wesentliche geschehen. Ich bin sicher, daß dieses Programm nicht nur für Susi von Nutzen ist!

37.1 Sie legen Ihr Geld an

Das Programm wollen wir ähnlich entwerfen, wie wir das für andere schon getan haben:

- Titel – damit Sie wissen, um was es geht.
- Auswahltabelle – damit Sie nach Belieben jede der vier Größen ermitteln können. Fangen wir damit an!

```
NEW
5 CLS
1Ø PRINT"      PROGRAMM ZUR KAPITALBERECHNUNG"
2Ø PRINT"=====================================
=====
3Ø PRINT
4Ø PRINT
5Ø PRINT"ZUR BERECHNUNG DES ENDKAPITALS  KE ..................
..................................... DRUECKEN SIE DIE TASTE 1"
6Ø PRINT"ZUR ERMITTLUNG DER LAUFZEIT  N ......................
..................................... DRUECKEN SIE DIE TASTE 2"
7Ø PRINT"ZUR ERMITTLUNG DES ZINSSATZES  Z ....................
..................................... DRUECKEN SIE DIE TASTE 3"

8Ø PRINT"ZUR ERMITTLUNG DES ENDKAPITALS  KE ..................
..................................... DRUECKEN SIE DIE TASTE 4"
```

Das wäre erledigt. Ich bin sicher, daß Sie sich stark genug fühlen, die Sperre gegen Falscheingabe selbst hinzuzufügen!

```
13Ø INPUT X
14Ø ON X GO TO 15Ø,25Ø,35Ø,45Ø
```

Ab Zeile 15Ø soll das Endkapital KE errechnet werden. Dazu müssen Sie den Computer zunächst die drei Größen:
Anfangskapital KA Zinssatz Z Laufzeit N abfragen lassen.

```
150 INPUT"WELCHEN BETRAG WOLLEN SIE ANLEGEN";KA
160 INPUT"WELCHEN ZINSSATZ BIETET IHNEN DIE BANK IN %";Z
170 INPUT"WIEVIEL JAHRE WOLLEN SIE DAS GELD ANLEGEN";N
```

Damit kennt der Computer die Größen, die zur Berechnung von KE erforderlich sind. Wir lassen ihn das erledigen nach der Formel für KE, die ich Ihnen vorhin aufgeschrieben hatte.

```
180 KE = KA*(1 + Z/100) ↑ N
```

Damit der Computer es nicht zu genaunimmt, geben wir noch die Formel für „Rundung auf zwei Stellen" ein:

```
190 KE = INT (KE*100+.5)/100
```

Und nun suchen wir noch eine gute Formulierung für die Ausgabe!

```
200 PRINT"WENN SIE ";KA;" DM ZU ";Z;" % ZINSEN FUER ";N;" JAHRE
          ANLEGEN, BETRAEGT IHR ENDKAPITAL KE ";KE;" DM"
210 PRINT"                    ==========================="
220 END
```

Der Rest des Programms besteht eigentlich aus einem „Thema mit Variationen". Sie sollten schon soviel Ehrgeiz haben, daß Sie sich daran selbst versuchen.

Für alle Fälle schreibe ich Ihnen den Rest – zum Nachschlagen – noch auf!

```
250 INPUT"WELCHEN BETRAG HABEN SIE ANGELEGT";KA
260 INPUT"ZU WELCHEM JAEHRLICHEN ZINSSATZ";Z
270 INPUT"WAS HAT IHNEN DIE BANK ZURUECKGEZAHLT";KE
280 N =(LOG(KE)–LOG(KA))/LOG(1+Z/100)
290 N = INT(N*100+.5)/100
300 PRINT"SIE MUESSEN IHR KAPITAL VON DM ";KA;" FÜR ";N;" JAHRE
    ANLEGEN"
310 PRINT"                                          ====="
320 END
350 INPUT"WELCHEN BETRAG HATTEN SIE ANGELEGT";KA
360 INPUT"WAS HAT IHNEN DIE BANK ZURUECKGEZAHLT";KE
370 INPUT"NACH WIEVIEL JAHREN";N
380 Z = 100*((KE/KA) ↑ (1/N)–1)
390 Z = INT(Z*100+.5)/100
400 PRINT"BEI DEN EINGEGEBENEN DATEN WAR DER ZINSSATZ ";Z;" %"
410 PRINT" = = =
```

```
480 KA = KE/((1+Z/100) ↑ N)
490 KA = INT(KA*100+.5)/100
500 PRINT"FUER ";KE;" DM ENDKAPITAL HAETTE IHR ANFANGSKAPITAL
                    ";KA;"
                    DM BETRAGEN MUESSEN"
510 PRINT" ==========================================
    ==="
520 END
```

37.2 Sie sehen Ihr Kapital wachsen

Mit dem folgenden Programm sollen Sie einmal zusehen, wie Ihr Sparkapital von Jahr zu Jahr durch den Zuwachs an Zinsen und Zinseszinsen anwächst. Das ist eine faszinierende Geschichte und kann Sie vielleicht zum Sparen (vielleicht für einen Drucker?) motivieren.

Als Ergebnis des Programms erwarten wir also, daß wir nicht nach einer Laufzeit N das Endkapital zu sehen bekommen; vielmehr wollen wir den Zuwachs von Jahr zu Jahr verfolgen.

```
NEW
5 CLS
10 PRINT" ..................... ZINSTAFEL ........................ "
20 PRINT" ========================================="
30 PRINT
40 INPUT"GEBEN SIE IHR ANFANGSKAPITAL EIN";KA
45 INPUT"WIE LANGE WOLLEN SIE DAS GELD FESTLEGEN";N
50 INPUT"WIEVIEL % ZINSEN GIBT IHNEN DIE BANK";Z
69 PRINT"LAUFZEIT IN JAHREN",,"ENDKAPITAL IN DM"
65 PRINT" ==========================================
    ==="
70 Y = 1
80 KE = KA*(1+Z/100) ↑ Y
90 KE = INT(KE*100+.5)/100
100 PRINT Y,,,KE
110 Y = Y+1
120 IF Y <= N THEN 80
130 PRINT"NACH ";N;" JAHREN BETRAEGT IHR ENDKAPITAL DM ";KE
140 S = KE – KA
150 PRINT"IHR ZUGEWINN BETRAEGT DANN ";S;" DM"
155 A = (S/KA) * 100
160 A=INT(A*100+.5)/100
```

```
170 PRINT"DAS SIND ";A;" % DES EINGESETZTEN KAPITALS VON DM ";KA
180 PRINT" ==========================================
    ==="
190 END
```

Damit Sie das richtig genießen können, zeige ich Ihnen auf der nächsten Seite einen „Ausdruck" für 25 Jahre – wohl wissend, daß Ihr Computer das nicht auf einmal auf den Schirm bringt.

```
................ ZINSTAFEL.....................................
==============================================
GEBEN SIE IHR ANFANGSKAPITAL EIN ? 1250
WIE LANGE WOLLEN SIE DAS GELD ANLEGEN ? 25
WIEVIEL % ZINSEN GIBT IHNEN DIE BANK ? 8
LAUFZEIT IN JAHREN                    ENDKAPITAL
```

1	1350
2	1458
3	1574.64
4	1700.61
5	1836.66
6	1983.59
7	2142.28
8	2313.66
9	2498.76
10	2698.66
11	2914.55
12	3147.71
13	3399.53
14	3671.49
15	3965.21
16	4282.43
17	4625.02
18	4995.02
19	5394.63
20	5826.2
21	6292.29
22	6795.68
23	7339.33
24	7926.48
25	8560.59

NACH 25 JAHREN BETRAEGT IHR ENDKAPITAL 856Ø.59 DM
IHR ZUGEWINN BETRAEGT DANN 731Ø.59 DM
DAS SIND 584.85 % DES EINGESETZTEN KAPITALS VON DM 125Ø

Noch ein paar Erklärungen gefällig?
Dieses Programm arbeitet wieder mit einem „Schleifenzähler", der in diesem Programm die Jahre zählt. Nachdem Sie in Zeile 45 schon die Gesamtlaufzeit mit N = 25 eingegeben hatten, müssen Sie sich nun zum Zählen der Jahre 1, 2, 3, 4, 5...25 eine andere Variable aussuchen. Ich habe hier Y gewählt.

In Zeile 7Ø wird Y auf 1 gesetzt. In Zeile 8Ø wird das Kapital berechnet. Zwar mit der gleichen Formel wie beim vorigen Programm. Aber HIER wollen Sie KE nach einem Jahr wissen. Und deshalb heißt es jetzt in der Formel KE = ... ↑ Y und nicht ... ↑ N!

In Zeile 12Ø ist eine weitere Form des „Austeigens" aus einer Schleife zu sehen. Diese Form hatten wir schon in den letzten Programmen verwendet. Aber hier folgt die Erklärung: Übersetzen Sie das wie folgt: Solange, als Y kleiner ist als N..."

In Zeile 14Ø ist der Zugewinn berechnet. Einfach Endkapital nach 25 Jahren minus eingesetztem Kapital. In Zeile 155 ist ausgerechnet, wieviel % des eingesetzten Kapitals von DM 1250 der Zugewinn von DM 7310.59 beträgt.

37.3 Der Computer verwaltet Ihre Schulden

Geld kann man nicht nur anlegen – man kann es sich auch leihen. Dann bekommen Sie keine Zinsen, sondern müssen welche bezahlen. Im folgenden Programm wollen wir uns einmal mit dieser Frage beschäftigen.

Es gibt noch einen anderen Grund, warum ich Ihnen dieses Programm vorstelle. Bei seiner „Entwicklung" bin ich zunächst in Schwierigkeiten geraten; das Ergebnis war unbefriedigend. Und weil es Ihnen sicher in Zukunft bei der Beschäftigung mit Ihrem Computer ebenso ergehen wird, will ich Ihnen die Entwicklungsgeschichte nicht vorenthalten.

Zunächst zu den Eingangsgrößen. Sie leihen sich bei Ihrer Bank Geld; wir nennen das die
Kreditsumme S.
Sie vereinbaren mit Ihrer Bank die Rückzahlung in festen Jahresraten über eine Laufzeit von N Jahren. Die Bank verlangt von Ihnen Z % Zinsen.
Die Jahresrate A, die Sie jährlich bis zur vollständigen Tilgung zurückzahlen müssen, ist A = S / N.
Die Zinsen, die Sie bezahlen müssen, betragen S * Z /1ØØ.
Nach dem ersten Jahr hat sich Ihre Schuld um eine Jahresrate A verringert; Sie müssen dann nur noch für S – A die vereinbarten Zinsen bezahlen. Und so setzt sich das im zweiten und dritten Jahr fort – solange, bis Ihre Schuld getilgt ist.

Mit diesen Daten habe ich das folgende Programm aufgebaut. Die erwähnten Schwierigkeiten hatte ich mit dem Ausstieg, den ich in den Zeilen 13Ø und 14Ø untergebracht hatte. Lassen Sie das Programm zunächst einmal so, wie es ist und sehen Sie, was dabei herausgekommen ist.

```
5 CLS
1Ø INPUT "WIE GROSS IST DER KREDIT IN DM"; S
2Ø INPUT "WIE LANG IST DIE LAUFZEIT IN JAHREN"; N
3Ø INPUT "WIE HOCH IST DER ZINSSATZ IN %"; Z
4Ø A = S/N
5Ø B = S * Z/1ØØ
55 CLS
6Ø PRINT "KREDITSUMME", "JAHRESZAHLUNG",
          "ZINS", "GESAMTBELASTUNG"
7Ø PRINT "DM)", "(DM)", "(DM)", "(DM/JAHR)"
8Ø PRINT" ==========================================
====="

9Ø PRINT
1ØØ PRINT S, A, B, B + A
11Ø S = S – A
12Ø B = S * Z/1ØØ
13Ø IF S <= A THEN 17Ø
14Ø IF S <= Ø THEN 16Ø
15Ø GO TO 1ØØ
16Ø END
17Ø PRINT "ES VERBLEIBT EINE RESTSCHULD VON"; S; "DM"
18Ø PRINT "RESTSCHULD MIT"; Z; "%"; "VERZINST, ERGIBT:";
    S * Z/1ØØ+S;"DM"
19Ø END
```

In Zeile 4Ø steht die Formel für die Jahreszahlung A; in Zeile 5Ø ist der Zinsbetrag in DM ausgerechnet, der bei dem Zinssatz Z im Jahr für die Kreditsumme S zu zahlen ist.

Nach der Überschrift in Zeilen 6Ø, 7Ø wollte ich in Zeile 1ØØ ausrechnen und hinschreiben lassen:

- Die jeweils noch verbleibende Kreditsumme S
- Die vereinbarte jährliche Rückzahlung A
- Den Zinsbetrag in DM, der für die jeweils verbleibende Kreditsumme zu bezahlen ist; hier mit der Variablen B benannt.
- Und schließlich meine jährliche Belastung – die sich ja aus der vereinbarten Rückzahlung A und den jeweils fälligen Zinsen zusammensetzt.

In Zeile 11Ø rechnet der Computer die gewünschten Größen schon für das zweite Jahr aus, in dem sich ja die Kreditsumme um eine Jahreszahlung verringert hat.

In Zeile 12Ø soll natürlich nur für diesen Betrag (nämlich S = S – A) der Zinsbetrag ausgerechnet werden. Und so geht das Programm weiter.

Kommen wir zu den Zeilen 13Ø und 14Ø, die den Ausstieg aus dem Programm bewerkstelligen sollen. Nach mehreren Versuchen, bei denen ich den Computer nur mit Gewalt stoppen konnte (nämlich mit BREAK), bin ich zu dieser Lösung gekommen. Sehen Sie sich an, was der Computer daraus gemacht hat!

```
WIE GROSS IST DER KREDIT IN DM? – 2Ø ØØØ
WIE LANG IST DIE LAUFZEIT IN JAHREN? – 5
WIE HOCH IST DER ZINSSATZ IN %? – 1Ø

KREDITSUMME      JAHRESZAHLUNG   ZINS      GESAMTBELASTUNG
(DM)             (DM)            (DM)      (DM/JAHR)
= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
2Ø ØØØ           4ØØØ            2ØØØ      6ØØØ
16 ØØØ           4ØØØ            16ØØ      56ØØ
12 ØØØ           4ØØØ            12ØØ      52ØØ
 8 ØØØ           4ØØØ             8ØØ      48ØØ
ES VERBLEIBT EINE RESTSCHULD VON 4ØØØ DM
RESTSCHULD MIT 1Ø % VERZINST, ERGIBT: 44ØØ DM
READY
> –
```

Zunächst habe ich geraten, warum der Computer die 5. Zeile (N = 5 Jahre) nicht mehr in die Tabelle geschrieben hat. Aber es war wenigstens richtig, was er ausgerechnet hat. Dann habe ich eine „krumme" Laufzeit eingegeben (N = 5.5). Das Ergebnis finden Sie unten.

```
WIE GROSS IST DER KREDIT IN DM? – 25 6ØØ
WIE LANG IST DIE LAUFZEIT IN JAHREN? – 5.5
WIE HOCH IST DER ZINSSATZ IN % – 12.4

KREDITSUMME      JAHRESZAHLUNG   ZINS      GESAMTBELASTUNG
(DM)             (DM)                      (DM/JAHR)
= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
256ØØ            4654.55         3174.4    7828.95
2Ø945.5          4654.55         2597.24   7251.78
16292Ø.9         4654.55         2.2Ø.Ø7   6674.62
11636.4          4654.55         1442.91   6Ø97.46
6981.82          4654.55         865.746   552Ø.29
```

```
ES VERBLEIBT EINE RESTSCHULD VON 2327.27 DM
RESTSCHULD MIT 12,4 % VERZINST, ERGIBT: 2615.86 DM
READY
> –
```

Hier hat nun der Computer für die „vollen" Jahre (5!) die Ergebnisse in die Tabelle geschrieben und mit dem Rest der Laufzeit (.5 Jahre!) die beiden letzten Zeilen ausgefüllt.

Erkennen Sie auf Anhieb, welcher Schnitzer jetzt in dem Programm steckt? Nein? Dann gehen Sie einmal zu einer Bank, die Ihnen für ein *halbes Jahr* den vollen Jahreszins abnimmt!!

Nach heftigem Nachdenken habe ich das Problem gelöst mit zwei Programmvarianten, die wir schon miteinander geübt haben:

- mit der UND-Schaltung
- mit dem bei INT... geübten Trennen von Dezimalzahlen.

Ich bitte, ggf. dort nachzulesen!

Sehen Sie zunächst das abgeänderte Programm; die Erklärungen dazu folgen unten!

```
130 N = N–1
140 R = N * 10
150 Y = INT (R/10)
160 E = R – (Y * 10)
170 IF (Y=0)* (E=0) THEN 190
180 IF (Y=0)* (E<>0) THEN 200
190 END
200 PRINT "RESTSCHULD", "RESTLAUFZEIT",
            "ZINSEN", "SCHLUSSZAHLUNG"
205 PRINT "(DM)", "(MONATE)", "(DM)", "(DM)"
210 PRINT" ==========================================
    ====="
215 K = ((S * Z / 100)/12) * E * 1.2
220 PRINT S, E * 1.2, K, S + K
230 END
```

Zeile 130: Ich darf daran erinnern, daß wir in Zeile 130 schon beim zweiten Durchlauf sind. Also im Jahr N – 1!
In den Zeilen 140 – 160 erfolgt das „Abtrennen".

Beispiel: N = 5	Beispiel: N = 5.5
...dann ist N – 1 = 4	N – 1 = 4.5
...dann ist R = 4Ø	R = 45
...dann ist Y = INT(R/1Ø)	
Y = 4	Y = 4
...dann ist E = R –(Y*1Ø)	
E = Ø	E = 5
Es ist also hier: Y = „Einer“	
E = „Zehntel“	

In den Zeilen 17Ø und 18Ø ist nun der Ausstieg aus dem Programm wie folgt programmiert:

17Ø IF (Y=Ø)*(E=Ø) THEN 19Ø

In Worten: Wenn Y = Ø UND (wenn) E = Ø, dann in Zeile 19Ø das Programm beenden!
Diese Bedingungen sind nur erfüllt, wenn N eine ganze Zahl ist; also 5 oder 8 oder 17.

18Ø IF (Y=Ø)*(E<>Ø) THEN 2ØØ

In Worten: Wenn Y = Ø UND (wenn) E *ungleich* Ø, dann in Zeile 2ØØ weitermachen.
Diese Bedingung ist nur erfüllt, wenn N eine *Dezimalzahl* ist; also 5,5 oder 3,6 oder 2Ø,9.

In Zeile 215 wird dann K ausgerechnet; das sind hier die Zinsen in DM für die verbleibende Restlaufzeit von (z. B.) .5 Jahren.

E*1.2 bedarf wohl noch der Erklärung? Nach dem oben Gesagten ist E die Anzahl der Zehntel-Jahre. Ein Jahr hat aber 12 Monate! Deswegen sind bei E = 5 (oder ½ Jahr!) für E*1.2 = 6 Monate die Zinsen fällig!

Einen Probelauf für dieses Programm sehen Sie unten. Ich war eigentlich ganz zufrieden damit. Übrigens: ich würde mich nicht wundern, wenn Sie auf Anhieb das Problem eleganter gelöst hätten. Sie wissen doch: 1Ø Programmierer ergeben 11 Programme. Mindestens!

WIE GROSS IST DER KREDIT IN DM? — 2ØØØØ
WIE LANG IST DIE LAUFZEIT IN JAHREN? – 5.5
WIE HOCH IST DER ZINSSATZ IN %? — 1Ø

```
KREDITSUMME      JAHRESZAHLUNG    ZINS        GESAMTBELASTUNG
(DM)             (DM)             (DM)        (DM/JAHR)
= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
20000            3636.36          2000        5636.36
16363.6          3636.36          1636.36     5272.73
12727.3          3636.36          1272.73     4909.09
 9090.91         3636.36          909.091     4545.45
 5454.55         3636.36          545.455     4181.82

RESTSCHULD       RESTLAUFZEIT     ZINS        SCHLUSSZAHLUNG
(DM)             (MONATE)         (DM)        (DM)
= = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = = =
1818.18          6                90.9092     1909.09
READY
>-
```

37.4 Was kostet Ihr Auto?

Mit dieser Frage ist nicht der Kaufpreis Ihres Autos gemeint. Gemeint sind auch nicht die Kosten eines Automobils, wie sie etwa ein Gewerbebetrieb ausrechnet, mit Abschreibung, Verzinsung usw. Dieses Programm soll Ihnen helfen, für Ihr privat genutztes Auto mehr zu bedenken als die Kosten für Kraftstoff und Öl. Der Kaufpreis spielt dabei keine Rolle – schließlich könnten Sie Ihr Auto geerbt haben. Und doch würde sein Betrieb etwas kosten.

Das Programm ist an sich einfach; es verwirrt zunächst durch die vielen verschiedenen Eingangsgrößen, die in die Berechnung der Betriebskosten einfließen. Das Ergebnis des Programms wird sein, daß Sie für Ihr Auto die Betriebskosten in DM/Jahr, DM/Monat und DM/km ausrechnen lassen. Im ersten Teil ermittelt der Computer die Treibstoffkosten pro Jahr; er fragt dazu nach:
Ihrer Fahrstrecke pro Jahr (J),
dem Verbrauch Ihres Autos in l/100 km (V)
und dem Preis für einen Liter Kraftstoff (P).
Aus diesen Größen wird in Zeile 40 TK errechnet. Dann wird die Kfz-Steuer ermittelt. Diese beträgt z. Zt. DM 14.40 pro angefangene 100 ccm Hubraum.

Konsequent fragt das Programm in Zeile 50 nach dem Hubraum H Ihres Autos. Nehmen wir zunächst an, dieser sei 1500 ccm.
60 H1 = H/100 ergibt dann 1500/100 = 15
70 H2 = INT (H1) ist dann ebenfalls 15
Der Vergleich in Zeile 80 ergibt dann NEIN – d. h. H1 – H2 IST 0 und NICHT UNGLEICH 0. Also geht es in Zeile 90 weiter:
90 S = H2 *14.4, damit ist die Kfz-Steuer errechnet.

Haben Sie aber ein Auto mit – z. B. – 1550 ccm Hubraum, dann ist

```
60 H1 = H/100          15.5
70 H2 = INT(H1)        15      somit
80 H1 – H2 IST UNGLEICH 0!
```

Also wird in Zeile 110 die Steuer S zu:

```
110 S = (H2*14.4)+14.4      in Worten:
      = 15 * 14.4 + 14.4 für die „angefangenen" 100 ccm!
```

Beachten Sie bitte bei diesem Programmpunkt, daß das Programm allgemein gültig sein muß; also auch ohne Änderung für ein Fahrzeug mit 1540 oder 1590 ccm den richtigen Wert für S liefern sollte.

Das Programm fragt dann in 200, ob Sie Wartung und Pflege für Ihr Fahrzeug selbst besorgen. Antworten Sie mit „JA", ist unterstellt: Sie kaufen nur Materialien wie Öl, Zündkerzen und ähnliche Verschleißteile und besorgen die notwendigen Arbeiten selbst.

```
Deshalb gilt dann: Mindestwartung      = DM 75.–/Jahr Material
                   Pflegedienst        = DM 0 (Sie machen es ja selbst!)
Antworten Sie mit „NEIN", dann soll gelten:
                   Mindestwartung      = DM 150/Jahr
                   Pflegedienst        = DM 350/Jahr
```

In Zeile 330 wird noch eine Pauschale für etwaige (Unfall)-Reparaturen mit RK = DM 600/Jahr berücksichtigt sowie in Zeile 370 nach den Kosten für die Versicherung VS – ebenfalls in DM/Jahr – gefragt. Und nach einigen Zwischensummen erfolgt dann aus allen Eingaben in den Zeilen 410 – 430 die Berechnung der eingangs verlangten Werte. Was kostet nun Ihr Auto wirklich?

```
  5 CLS
 10 INPUT "WIEVIEL KM FAHREN SIE IM JAHR"; J
 20 INPUT "WIEVIEL LITER VERBRAUCHT IHR AUTO AUF 100 KM"; V
 30 INPUT "WAS KOSTET 1 LITER KRAFTSTOFF IN DM"; P
 40 TK = (J/100) * V * P
 50 INPUT "WELCHEN HUBRAUM IN CCM HAT IHR AUTO"; H
 60 H 1 = H/100
 70 H 2 = INT (H 1)
 80 IF H 1 – H 2 <> 0 THEN 110
 90 S = H 2 * 14.4
 95 GOSUB 200
100 GO TO 360
110 S = (H 2 * 14.4) + 14.4
```

```
120  GOSUB 200
130  GO TO 360
200  PRINT "MACHEN SIE WARTUNG UND PFLEGE SELBST?"
210  PRINT "WENN JA: DRUECKEN SIE 1!"
220  PRINT "WENN NEIN: DRUECKEN SIE 0!"
230  INPUT Q
240  IF Q = 1 THEN 300
250  IF Q = 0 THEN 320
260  CLS
270  FOR X = 1 TO 1000 : NEXT
280  PRINT "FALSCHEINGABE: 0 ODER 1 DRUECKEN!"
290  GO TO 200
300  MW = 75 : PD = 0
310  GO TO 330
320  MW = 150 : PD = 350
325  GO TO 330
330  RK = 600
340  Z = RK + MW + PD
350  RETURN
360  Z 1 = TK + S + Z
370  INPUT "WELCHEN BETRAG ZAHLEN SIE PRO JAHR FUER
     HAFTPFLICHT"; VS
380  Z 2 = Z 1 + VS
390  CLS
400  PRINT "IHRE BETRIEBSKOSTEN BETRAGEN:"
410  PRINT Z 2; "DM PRO JAHR"
420  PRINT Z 2/12; "DM PRO MONAT"
430  PRINT Z 2/J; "DM PRO KILOMETER"
440  END
```

38 Der Computer als Sortiermaschine

Wie lesen Sie ein Adreßbuch, wenn Sie dort den Namen „Weber" suchen? Vermutlich blättern Sie zunächst das Buch „grob" durch, bis Sie die Seiten mit dem Buchstaben „W..." gefunden haben. Dann werden Sie etwas „feinfühliger"; Sie finden WA... – und wissen, daß Sie noch weiterblättern müssen. Schließlich sind Sie bei WE..., suchen weiter bis WEB... usw. usw.

Wenn Sie dem Computer eine Liste mit – sagen wir, Familiennamen – anbieten, die nicht geordnet ist, dann kann der Computer diese für Sie in emsiger interner Arbeit umstellen und Ihnen auf Wunsch in alphabetischer Reihenfolge geordnet wieder ausgeben.

Aber was heißt schon „auf Wunsch"? „Auf entsprechenden Befehl" – das wäre wohl richtig! Der Computer geht dabei ähnlich vor, wie Sie oben bei der Suche nach dem Namen „Weber". Von der „Ordnungsarbeit" des Computers merken Sie nichts, Sie sehen später nur die geordnete Liste.

Mit dieser Problematik wollen wir uns jetzt beschäftigen. Das Thema ist recht knifflig – weshalb es auch so ziemlich am Ende des Buches erscheint.

Aber wir werden es schon schaffen! Sehen Sie sich zunächst einmal das folgende Programm im ganzen an:

```
NEW
 5 CLS
1Ø INPUT"ANZAHL DER ZU ORDNENDEN NAMEN";A
2Ø FOR X = 1 TO A
3Ø INPUT N$(X)
4Ø NEXT X
```

Mit den Zeilen 1Ø...4Ø geben Sie dem Computer die noch ungeordnete Liste der Namen ein. Dieser „Eingabeteil" ist für Sie nicht neu; wir haben ihn schon in anderen Programmen geübt. Hängen wir gleich den „Ausgabeteil" hinten an!

```
15Ø FOR X = 1 TO A
16Ø PRINT N$(X)
17Ø NEXT X
```

Mit den Programmzeilen 1Ø...15Ø könnten Sie die Liste eingeben; der Computer würde sie in der gleichen Form – also ungeordnet – wieder ausgeben.

Jetzt folgt das Programm, das den Computer zum gewünschten Sortieren befähigt.

```
 50 Z = A - 1
 60 FOR X = 1 TO A - 1
 70 FOR Y = 1 TO Z

 80 IF N$(X) < N$(X+Y) THEN 120

 90 H$ = N$(X)
100 N$(X) = N$(X+Y)
110 N$(X+Y) = H$

120 NEXT Y
130 Z = Z - 1
140 NEXT X
```

So, das wär's. Bevor es ans Erklären geht, sollen Sie erst einmal ein Erfolgserlebnis haben. Starten Sie dieses Programm! Geben Sie in Zeile 10 für A 4 ein. Und nun nacheinander:

HORST
KLAUS
EMIL
JUERGEN

Augenblicklich „sortiert" der Computer und gibt in alphabetischer Reihenfolge aus:

EMIL
HORST
JUERGEN
KLAUS

Lassen Sie das Programm stehen! Wie Sie erkennen, besteht das Sortierprogramm aus 4 voneinander optisch abgesetzten Blocks. In den Zeilen 50...70 werden zwei FOR...NEXT-Schleifen und eine Zählschleife (Zeile 50) begonnen. In den Zeilen 120...140 werden die Schleifen beendet.

Nun zum dritten „Block" – er besteht nur aus der Zeile 80. Wie Sie an der Formulierung des Befehls erkennen, erfolgt dort ein Vergleich der beiden Strings N$(X) mit N$(X+Y). Der Computer soll nachschauen, ob der erste String KLEINER ist als der zweite.

Erinnern Sie sich an die einleitenden Kapitel? Dort hatte ich Ihnen erläutert, daß der Computer in seinem „Inneren" weder mit 1, 2, 3... noch mit A, B, C... etwas anfangen kann. Er benötigt alle eingegebenen Zeichen in einer für ihn speziellen „Verschlüsselung". Für unseren weiteren Fortschritt im „Sortierprogramm" brauchen wir nicht in die Maschinensprache einzusteigen. Es genügt, wenn ich Ihnen versichere, daß zunächst alle Buchstaben in einen Zahlencode umgewandelt werden. Das geschieht nach einer weltweit anerkannten Norm, dem sogenannten „ASCII-Code".

ASCII steht für "American Standard Code für Information Interchange" und das heißt: Amerikanischer Code für Informations-Austausch. In der *Tabelle* unten habe ich Ihnen einen Ausschnitt daraus aufgeschrieben.

Auszug aus dem ASCII-Code

A) Buchstaben

Dezimaler Code	Zeichen	Dezimaler Code	Zeichen
65	A	78	N
66	B	79	O
67	C	8Ø	P
68	D	81	Q
69	E	82	R
7Ø	F	83	S
71	G	84	T
72	H	85	U
73	I	86	V
74	J	87	W
75	K	88	X
76	L	89	Y
77	M	9Ø	Z

B) Ziffern und Zeichen

Dezimaler Code	Zeichen	Dezimaler Code	Zeichen
48	Ø		
49	1		
5Ø	2	32	
51	3	33	!
52	4	34	"
53	5	35	#
54	6	36	$
55	7	37	%
56	8	38	&
57	9	39	'
		4Ø	(

Achtung! Code 32 ist ein *Leerzeichen*! (BLANK)

Hier ist wichtig, daß Sie daraus erkennen: Jedes Zeichen im Alphabet hat eine *kleinere* Code-Nummer als das folgende.

A hat den Code 65
B hat den Code 66
.
.
.
Y hat den Code 89
Z hat den Code 90 usw.

Jetzt werden Sie verstehen, was in der Zeile 80 geschieht!
Der Computer „schaut nach", ob die Code-Nummer des ersten Buchstabens des Strings N$(X) KLEINER ist als die Code-Nummer des ersten Buchstabens im String N$(X+Y).
Beispiel: HORST < KLAUS bedeutet: Code für H < Code für K?

Und wenn der Code nicht kleiner ist?
Dann geht es laut Programm in unserem vierten Programmblock – in den Zeilen 90...110 – weiter. Dort werden einfach die beiden Strings miteinander vertauscht, „herumgedreht".
Stand etwa in Zeile 80 KLAUS < HORST
dann wird daraus in Zeile 110 HORST < KLAUS
Sie brauchen sich nicht zu genieren! Ich habe das auch nicht sofort begriffen. In solchen Fällen ist es immer gut, sich über ein überschaubares Beispiel dem Problem zu nähern. Das wollen wir mit dem folgenden Zwischenprogramm einmal tun!

Die „Vertauschungsformel" heißt:

```
 90 H$ = N$(X)
100 N$(X) = N$(X+Y)
110 N$(X+Y) = H$
```

Zur Vereinfachung setzen wir	H$	zu H
	N$(X)	zu N
	N$(X+Y)	zu M

Für N und M belegen wir nun Zahlenwerte; und zwar für N = 10, für M = 20.
Und nun zu unserem „Demo-Programm":

```
1000 N = 10
1010 M = 20
1020 PRINT"N =",,"M ="
1030 PRINT N,,M
```

Jetzt wird darauf die „Vertauschungsformel" angewendet:

```
1040 H = N
1050 N = M
1060 M = H
```

erkennen Sie, daß H als Hilfsvariable später einfach „unter den Tisch fällt"?

Nun sind Sie neugierig, was dabei herauskommt? Also:

```
1070 PRINT"NACH DEM VERTAUSCHEN IST"
1080 PRINT"N =",,"M ="
1090 PRINT N,,M
```

Und nun: ACHTUNG!! RUN 1000

Haben Sie das Ergebnis erwartet?

```
N =                                    M =
10                                     20
NACH DEM VERTAUSCHEN IST
N =                                    M =
20                                     10
```

Und nun halte ich Sie für genügend präpariert, daß Sie das komplette Sortierprogramm Schritt für Schritt nachvollziehen können.

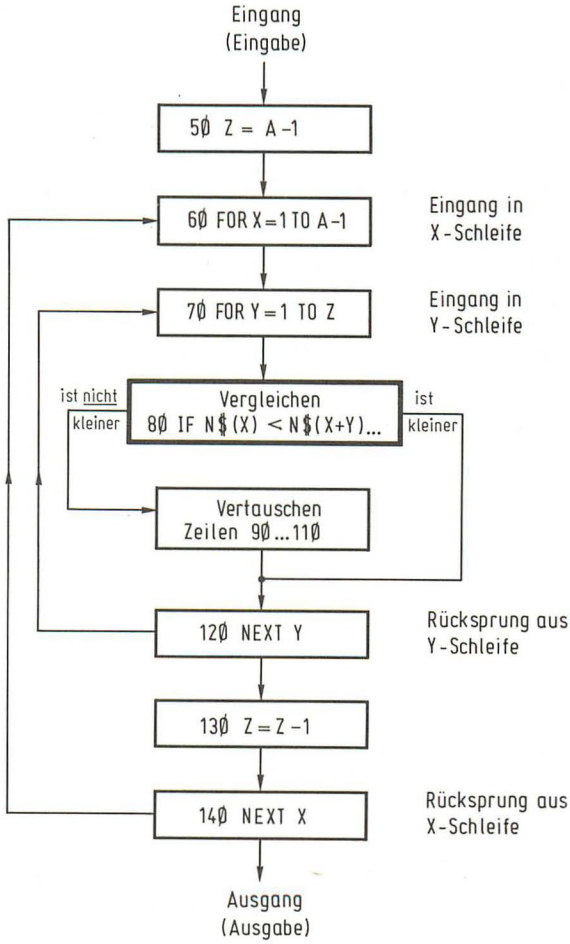


Abb. 30 So arbeitet das Sortierprogramm

Hier zunächst eine graphische Darstellung des Programms, aus der Sie sehr gut erkennen können, wie die beiden FOR...-NEXT-Schleifen ineinandergeschachtelt sind. Ziehen Sie die Abbildung beim Verfolgen der Beschreibung immer wieder zu Rate (*Abb. 30*)!

Erster Durchlauf:

Eingangsgrößen:

N $ (1) = HORST
N $ (2) = KLAUS
N $ (3) = EMIL
N $ (4) = JUERGEN
A = 4
Z = A–1; Z = 3

Start des Programms:

5Ø Z = A–1	Setzen des „Schleifen-Zählers" Z auf 3
6Ø FOR X = 1 TO A–1	Beginn der X-Schleife. Diese soll von 1 bis A–1, also von 1–3 laufen.
7Ø FOR Y = 1 TO Z	Beginn der Y-Schleife. Diese soll von 1 bis Z, also von 1–3 laufen.

8Ø IF N $ (X) < N $ (X+Y) THEN 12Ø

In Zeile 8Ø erfolgt der VERGLEICH. Für den ersten Durchlauf der Y-Schleife =
– 8Ø IF N $ (1) < N $ (2) THEN...
laut Eingabe ist:
N $ (1) = HORST
N $ (2) = KLAUS

Der Vergleich ergibt:
HORST ist KLEINER als KLAUS deshalb Sprung zu 12Ø
12Ø NEXT Y
d. h., zurück zu Zeile 7Ø, jedoch mit Y = 2!
8Ø IF N $ (X) < N $ (X + Y) THEN 12Ø
Das Programm ist immer noch in der X-Schleife!
Deshalb ist: X = 1; X + Y = 3
– 8Ø IF N $ (1) < N $ (3) THEN 12Ø
– Laut Eingabe ist:
N $ (1) = HORST
N $ (3) = EMIL

Der Vergleich ergibt:
HORST ist NICHT KLEINER als EMIL, deshalb weiter mit 9Ø

9Ø H $ (X) = N $ (X)	H $ ist eine Hilfsvariable. Diese erhält den „Wert" N $ (1) = HORST

1ØØ N $ (X) = N $ (X + Y)	Hier vertauschen die Variablen Ihren „Wert". N $ (1) wird zu N $ (3)
11Ø N $ (X + Y) = H $	Also: N $ (1) wird EMIL, N $ (3) wird zu HORST
12Ø NEXT Y	Wieder zurück in die Y-Schleife. Y wird damit 3 Achtung: Damit ist Y = Z; (Y = 3) Dies ist der letzte Durchlauf in der Y-Schleife!

8Ø IF N $ (X) < N $ (X + Y) THEN 12Ø
Das Programm ist immer noch in der X-Schleife!
– 8Ø IF N $ (1) < N $ (4) THEN 12Ø
– laut Eingabe/Umwandlung ist:
N $ (1) = EMIL
N $ (4) = JUERGEN
Der Vergleich ergibt:
EMIL ist KLEINER als JÜRGEN
Da aber mit Y = 3 diese Schleife gemäß Vereinbarung in: 7Ø FOR Y = 1 TO 3 zu Ende ist, wird diese verlassen zu

13Ø Z = Z – 1	Z war 3, wird zu 2
14Ø NEXT X	RÜCKSPRUNG in die X-Schleife zu deren zweitem Durchgang.

Der erste Durchlauf ist also beendet. Wie haben sich für die nächste Runde die Eingangsgrößen verändert?

Zweiter Durchlauf:
Eingangsgrößen:
N $ (1) = EMIL
N $ (2) = KLAUS
N $ (3) = HORST
N $ (4) = JÜRGEN
X = 2
Y = 1 – weil die Y-Schleife wieder von vorn beginnt!

6Ø FOR X = 2 TO A – 1	X = 2; da 2. Durchlauf!
7Ø FOR Y = 1 TO Z	Z ist jetzt 2

8Ø IF N $ (X) < N $ (X + Y) THEN 12Ø
In Zeile 8Ø erfolgt der Vergleich
8Ø IF N $ (2) < N $ (3) THEN 12Ø
N $ (2) ist KLAUS
N $ (3) ist HORST
Der Vergleich ergibt:
KLAUS ist NICHT KLEINER als HORST; deshalb:

9Ø H $ = N $ (X)	H $ erhält jetzt den „Wert" KLAUS
1ØØ N $ (X) = N $ (X + Y)	Die Variablen vertauschen ihren „Wert"

```
110 N $ (X + Y) = H $        Also: N $ (2) wird HORST
                                   N $ (3) wird KLAUS
120 NEXT Y                   Y wird jetzt zu 2, da 2. Lauf der Y-Schleife
```

80 IF N $ (X) < N $ (X + Y) THEN 120
In Zeile 80 erfolgt der Vergleich
80 IF N $ (2) < N $ (4) THEN 120
N $ (2) ist: HORST
N $ (4) ist: JÜRGEN

Der Vergleich ergibt:
HORST ist KLEINER als JÜRGEN, deshalb zu

```
120...                       hier ist vereinbart worden:
                             ...FOR Y = 1 TO Z; FOR Y = 1 TO 2
```

Y = 2 war aber bereits erreicht. Deshalb erfolgt

```
130 Z = Z – 1                Z war 2, wird 1
140 NEXT X                   RÜCKSPRUNG in die X-Schleife
                             zu deren drittem Durchgang.
```

Zu Beginn des dritten Durchlaufs ist nun:

N $ (1) = EMIL
N $ (2) = HORST
N $ (3) = KLAUS
N $ (4) = JÜRGEN
X = 3
Y = 1 weil die Y-Schleife wieder von vorn beginnt.

```
60 FOR X = 3 TO A – 1        – also nur noch ein Durchlauf
70 FOR Y = 1 TO Z            – Z ist jetzt 1
```

80 IF N $ (X) < N $ (X + 1) THEN 120
In Zeile 80 erfolgt der Vergleich!
80 IF N $ (3) < N $ (4) THEN 120
N $ (3) = KLAUS
N $ (4) = JÜRGEN
Der Vergleich ergibt:
KLAUS ist NICHT KLEINER als JÜRGEN, deshalb zu:

```
 90 H $ = N $ (X)            H $ wird zu KLAUS
100 N $ (X) = N $ (X + Y)    Die Variablen vertauschen Ihren „Wert“
110 N $ (X + Y) = H $        Also: N $ (3) = JÜRGEN
                                   N $ (4) = KLAUS
120 NEXT Y                   Y-Schleife ist beendet!
```

```
130 Z = Z - 1                Z wird zu 0
140 NEXT X                   X-Schleife ist beendet!
```

Also: DAS SPIEL IST AUS!

Nach dem 3. Durchlauf war: N $ (1) = EMIL
N $ (2) = HORST
N $ (3) = JÜRGEN
N $ (4) = KLAUS

Das ist die Liste, korrekt alphabetisch sortiert.

Eine richtig sortierte Liste von Namen oder Begriffen wird nicht nur nach der alphabetischen Reihenfolge des ersten Buchstabens sortiert. Wie Sie sich beim Blick in jedes Telefonbuch überzeugen können, wird nach dem ersten der zweite Buchstabe, danach der dritte usw. auf alphabetische Reihenfolge hin untersucht. Überprüfen Sie, ob das Programm auch dazu in der Lage ist!

Geben Sie ein:

N = 5
ACHERN
AHNSEN
AALEN
AHLEN
AACHEN

Das Programm wird daraus machen:

AACHEN
AALEN
ACHERN
AHLEN
AHNSEN

Auch Zahlen-Listen können Sie durch den Computer sortieren lassen. Dafür verwenden Sie im Prinzip das gleiche Programm; ich stelle es Ihnen unten – für Zahlen abgewandelt – vor.

```
10 INPUT"WIEVIELE ZAHLEN";A
20 FOR X = 1 TO A
30 INPUT N(X)                                   EINGABE
40 NEXT X
```

```
50 Z = A - 1
60 FOR X = 1 TO A - 1
70 FOR Y = 1 TO Z
80 IF N(X) < N(X+Y) THEN 120
90 H = N(X)                                     SORTIEREN
```

```
100 N(X) = N(X+Y)
110 N(X+Y) = H
120 NEXT Y
130 Z = Z - 1
140 NEXT X
```

```
150 FOR X = 1 TO A
160 PRINT N(X)                    AUSGABE
170 NEXT X
```

Was haben Sie dazugelernt?

- Wie man Namen – oder, allgemein – Strings nach dem Alphabet sortiert.
- Wie man Zahlen sortiert.
- Wichtig – auch für die Programmentwicklung allgemein – ist die im Sortierprogramm steckende „Austauschformel", mit der die Werte zweier numerischen oder Stringvariablen ausgetauscht werden.

```
H$ = N$(I)          H = N(I)
N$(I) = N$(I + Y)   N(I) = N(I + Y)
N$(I+Y) = H$        N(I + Y) = H
```

Es gibt übrigens eine Reihe von verschiedenen Sortierverfahren; eines davon werden Sie später noch kennenlernen.
Wenn Sie in den Sortierprogrammen einmal mehr als 4 oder 5 Begriffe sortieren lassen, werden Sie bemerken, daß zwischen dem Start des Programms und der Ausgabe der Computer einen Augenblick „zögert". Diese deutlich erkennbare Zeit benötigt er zum Sortieren. Die erwähnten verschiedenen Sortierverfahren unterscheiden sich deshalb hauptsächlich in der notwendigen Ausführungszeit.

39 Der Computer zum Spielen

Gerade für Hobby-Computer gibt es eine Fülle von „Unterhaltungsprogrammen". Ich will hier über Wert oder Sinn solcher Programme oder über die Beschäftigung damit nicht polemisieren – ich finde, man wird sie sehr schnell leid – sondern Ihnen lieber an Beispielen zeigen, wie Sie selbst solche Programme aufstellen können. Vielleicht macht Ihnen das mehr Spaß!

Die meisten derartigen Programme arbeiten mit Zufallszahlen; Sie setzen dafür den „Zufallsgenerator" Ihres Computers ein.

Ein weiteres Element der Programme ist die IF...THEN...-Entscheidung, mit der Sie ja hinreichend vertraut sind.

39.1 Das Zahlen-Ratespiel

In diesem Programm „zieht" der Computer aus dem Zufallsgenerator eine beliebige Zahl, die Ihnen zunächst verborgen bleibt. Sie sollen durch entsprechende Eingabe die gezogene Zahl „erraten". Treffen Sie die richtige Zahl, gibt der Computer diese aus und sagt Ihnen gleichzeitig, wieviele Eingaben Sie bis zum richtigen Ergebnis benötigt haben.

```
NEW
5 CLS
10 PRINT"ICH – IHR COMPUTER – DENKE MIR EINE GANZE ZAHL"
20 PRINT"DIE ZAHL LIEGT ZWISCHEN 1 UND 100"
30 PRINT
40 PRINT"SIE SOLLEN DIESE ZAHL SO SCHNELL WIE MOEGLICH ERRATEN"
50 R = RND(100)
60 Y = 0
70 PRINT
80 PRINT"ICH BIN BEREIT!"
90 PRINT
100 INPUT"WAS IST IHR VORSCHLAG";X
110 Y = Y + 1
120 IF X = R THEN 145
130 IF X > R THEN 250
140 GO TO 270
145 CLS
```

```
150 PRINT"ICH GRATULIERE ! SIE HABEN'S ERRATEN !"
160 PRINT"SIE HATTEN ";Y;" VERSUCHE !"
170 PRINT"WOLLEN SIE NOCH EINMAL RATEN"
180 PRINT"WENN JA, DRUECKEN SIE.................1"
190 PRINT"WENN NEIN, DRUECKEN SIE..........2"
200 INPUT L
210 IF L = 1 THEN 240
215 CLS
220 PRINT"SCHADE ! HAT ES IHNEN NICHT GEFALLEN ?"
225 END
240 FOR A = 1 TO 500 : NEXT
245 CLS
248 GO TO 50
250 PRINT"ZU HOCH ! VERSUCHEN SIE ES NOCH EINMAL !"
260 GO TO 100
270 PRINT"ZU NIEDRIG ! BITTE NOCH EINMAL !"
280 GO TO 100
```

Ich bin sicher, daß Sie keine Schwierigkeiten haben, dieses Programm zu durchschauen.

Damit Sie sich bei Ihrem Computer nicht blamieren – weil Sie zu lange raten – will ich Ihnen einen Weg zeigen, wie man in solchen und ähnlichen Programmen am schnellsten zum Ziel kommt. Nehmen Sie dazu an, der Computer habe sich die Zahl 86 „gedacht".

Ihre Eingabe	Die Ausgabe des Computers
50	ZU NIEDRIG ! BITTE NOCH MAL !
75	ZU NIEDRIG ! BITTE NOCH MAL !
88	ZU HOCH ! VERSUCHEN SIE ES NOCH EINMAL !
81	ZU NIEDRIG ! BITTE NOCH EINMAL !
86	ICH GRATULIERE ! SIE HABEN'S ERRATEN !
	SIE HATTEN 5 VERSUCHE !

Mit meiner ersten Eingabe habe ich mich bewußt für die MITTE der 100 möglichen Zahlen entschieden. Nach der Antwort des Computers gab es nur noch 50 Möglichkeiten. Wieder habe ich mit 75 die Mitte gewählt usw. Die letzte Eingabe 86 war purer Zufall.

Blicken Sie noch einmal auf die Zeilen 180 bis 225 des Programms. Hier „bewertet" der Computer Ihre Eingabe für L, um je nach der eingegebenen Zahl

eine Entscheidung für „Weitermachen" oder „Abbruch" zu treffen. Es gibt für derartige Entscheidungen zahlreiche Programmvarianten; die kürzestmögliche ist folgende:

```
1000 INPUT"L=";L
1010 IF L = 1 THEN 1030
1020 PRINT"FALSCH"
1025 END
1030 PRINT"RICHTIG"
.........
.........
```

Einen vergleichbaren Zusammenhang finden Sie in den Zeilen 120...145. Dort soll der Computer Ihre Eingabe (X) mit der aus dem Zufallsgenerator gezogenen Zahl (R) vergleichen. Dafür gibt es drei Möglichkeiten, nämlich:

X = R
X > R
X < R

Das Programm prüft nur zwei dieser Möglichkeiten (Zeilen 120, 130). Treffen beide NICHT zu, dann kann nur die dritte übrigbleiben und der Computer handelt mit X < R weiter im Programm. Auch diese Programmvariante werden Sie noch oft benötigen; Sie sollten sich diese also noch einmal genau ansehen und einprägen!

39.2 Der Computer bringt Ihre Party in Schwung

Sie werden jetzt ein Programm schreiben, das Ihre nächste Party ganz schön in Stimmung bringt. Und zwar wird der Computer – ohne daß Sie dabei in Erscheinung treten – die Namen der geladenen Gäste erraten. Als Computerfachmann wissen Sie natürlich, daß er das ohne Ihre Hilfe nicht kann. Sie präparieren ihn für seine Aufgabe, bevor die Gäste erscheinen. Für das Programm benötigen Sie einige Informationen über Ihre Gäste, die Sie dem Computer anvertrauen und die er mit den Eingaben der Gäste vergleicht.

Das Prinzip dieses einfachen – aber wirkungsvollen – Spiels erkennen Sie auch, wenn es nur für wenige Gäste programmiert wird; Sie können das Programm gern selbst für eine „Riesenparty" abwandeln. Zunächst schauen Sie sich die Liste Ihrer Gäste an. Darin steht außer dem Namen der Geladenen noch ein Minimum an Zusatzinformationen. Und das ist Ihre Gästeliste:

1. Klaus Müller, Ihr Duzfreund, noch Junggeselle.
2. Erika Meier, unverheiratet, „hat was" mit Klaus Müller
3. Gerd Weber, verheiratet mit Gast 4, von Beruf Ingenieur, arbeitet in einem Industriebetrieb.

4. Gerda Weber, Frau von Gast 3, Hausfrau, 2 Kinder.
5. Heinrich Bach, verheiratet mit Gast 6, selbständiger Handwerker.
6. Helga Bach, verheiratet mit Gast 5, keine Kinder, Lehrerin.

Aus diesem „Wissen" müssen Sie Fragen formulieren, die der Computer Ihren Gästen stellen soll, um aus deren Antworten eindeutig auf den Namen des Gastes zu schließen. Diese Fragen sollen natürlich so unverfänglich wie möglich sein! Ich nenne Ihnen ein paar Auswahlkriterien, die für das Erraten Ihrer Gäste ausreichen.

A) Der Gast ist entweder ein Mann oder eine Frau.
B) Der Gast ist entweder verheiratet oder ledig.
C) Der Gast ist selbständig oder unselbständig tätig.
D) Der Gast hat Kinder oder er hat keine Kinder.

Beginnen Sie Ihr Programm mit einer hübschen Einleitung!

```
NEW
5 CLS
1Ø PRINT
2Ø PRINT
3Ø PRINT" G U T E N  A B E N D "
4Ø PRINT"ICH MOECHTE IHNEN EIN SPIEL VORSCHLAGEN !"
5Ø PRINT
6Ø PRINT"ICH WERDE ERRATEN, WER SIE SIND UND IHNEN DAZU FRAGEN
   STELLEN"
7Ø PRINT
8Ø PRINT"WENN SIE MITSPIELEN WOLLEN:
..................................... DRUECKEN SIE DIE TASTE 1"
9Ø PRINT"WENN NICHT
..................................... DRUECKEN SIE DIE TASTE 2"
1Ø5 PRINT"DANACH JEWEILS DIE TASTE 'ENTER' "
11Ø INPUT Q
```

So weit, so gut. Wir bauen zunächst eine Sperre auf für den Fall, daß Ihre noch computerunerfahrenen Gäste falsche Tasten drücken!

```
116 IF Q < 1 THEN 1ØØØ
118 IF Q > 2 THEN 1ØØØ
1ØØØ CLS
1ØØ5 PRINT"SIE HABEN EINE FALSCHE TASTE GEDRUECKT !"
1Ø1Ø PRINT"BEGINNEN WIR NOCH MAL VON VORN"
```

Und damit sich der Gast von seinem Schreck erholen kann, bauen Sie eine kurze Pause ein!

```
1020 FOR X = 1 TO 500 : NEXT
1030 GO TO 5
```

Setzen Sie das Programm fort mit:

```
115 CLS
120 IF Q = 1 THEN 200
130 IF Q = 2 THEN 135
135 PRINT"DAS IST ABER SCHADE ! ICH WUENSCHE IHNEN TROTZDEM
          EINEN SCHOENEN ABEND !"
140 END
```

Nun sollten Sie ein „Sieb" aus Fragen aufbauen, das alle Ihre Gäste passieren müssen. Das erste und einfachste Unterscheidungsmerkmal ist wohl:
MANN oder FRAU

```
200 PRINT"MEINE ERSTE FRAGE:
          SIND SIE EIN  M A N N  ODER EINE  F R A U ?"
205 PRINT"DRUECKEN SIE DIE TASTE 1 FUER  M A N N !"
210 PRINT"DRUECKEN SIE DIE TASTE 2 FUER  F R A U !"
220 INPUT A
230 IF A = 1 THEN 280
240 IF A = 2 THEN 500
250 IF A < 1 THEN 1000
260 IF A > 2 THEN 1000
280 CLS
285 PRINT" GUTEN ABEND MEIN HERR!
          SIND SIE VERHEIRATET?"
290 PRINT"JA      =      1      ;      NEIN      =      2"
```

Wenn der angesprochene Herr jetzt – hoffentlich wahrheitsgemäß – mit 2 antwortet, dann wissen Sie, daß es Ihr Freund Klaus sein muß. Denn er ist der einzige unverheiratete Mann auf Ihrer Party.

```
300 INPUT B
310 IF B = 1 THEN 380
320 IF B = 2 THEN 350
330 IF B < 1 THEN 1000
340 IF B > 2 THEN 1000
350 CLS
355 PRINT"HALLO, KLAUS! ERIKA IST AUCH DA!"
360 GO TO 1020
```

War der Mann verheiratet, fragen Sie nach seinem Beruf.

```
380 CLS
385 PRINT"MEINE NÄCHSTE FRAGE:
```

```
            GELTEN SIE FUER DAS FINANZAMT ALS SELBSTÄNDIGER?"
390 PRINT"JA      =      1      ;      NEIN      =      2"
400 INPUT C
410 IF C = 1 THEN 470
420 IF C = 2 THEN 445
430 IF C < 1 THEN 1000
440 IF C > 2 THEN 1000
```

Diese Frage kann Herr Weber nur mit 2, Herr Bach nur mit 1 beantworten!

```
445 CLS
450 PRINT"ICH HAB'S! SIE SIND HERR WEBER!"
460 GO TO 1020
470 CLS
480 PRINT"ICH BEGRUESSE SIE, HERR BACH!"
490 GO TO 1020
```

Damit sind alle geladenen Herren identifiziert. Wenden Sie sich den Damen zu!

```
500 CLS
505 PRINT"GUTEN ABEND, MEINE DAME!
            SIND SIE VERHEIRATET?"
510 PRINT"JA      =      1      ;      NEIN      =      2"
520 INPUT D
530 IF D = 1 THEN 585
540 IF D = 2 THEN 565
550 IF D < 1 THEN 1000
560 IF D > 2 THEN 1000
```

Nun kann sich mit 2 nur Erika zu erkennen gegeben!

```
565 PRINT"HALLO, ERIKA! PASS' GUT AUF KLAUS AUF!"
580 GO TO 1020
585 CLS
590 PRINT"HABEN SIE KINDER, MEINE DAME?"
600 INPUT E
610 IF E = 1 THEN 670
620 IF E = 2 THEN 650
630 IF E < 1 THEN 1000
640 IF E > 2 THEN 1000
650 CLS
655 PRINT"GUTEN ABEND, FRAU BACH!
            ICH WUENSCHE VIEL VERGNUEGEN!"
660 GO TO 1020
670 CLS
```

```
675 PRINT"GUTEN ABEND, FRAU WEBER! WAS MACHEN DIE KINDER?"
680 GO TO 1020
```

Ich bin ganz sicher, daß Ihre Gäste über dieses einfache Programm ganz verblüfft sind. Und Ihr Ansehen als Computerfachmann steigt sehr.

39.3 Sie spielen gegen Ihren Computer

Wenn Sie der Computerei treu bleiben, werden Sie früher oder später nicht mehr nur mit eigenen, sondern auch mit fremden Programmen arbeiten.

Es gibt viele Möglichkeiten, an fremde Programme zu kommen. Sie können sie mit Gleichgesinnten austauschen oder Fachzeitschriften und speziellen Büchern entnehmen. Schließlich gibt es Firmen – „Software-Häuser" –, die sich auf die Herstellung und den Vertrieb von Computerprogrammen spezialisiert haben und Ihnen diese als „Listing" oder auf Datenträgern – wie Magnetbandkassetten oder Floppy – anbieten.

Daß die Muttersprache Ihrer Basic-Befehle Englisch ist, haben Sie inzwischen gelernt. Vielfach sind auch die fremden Programme, die aus den oben beschriebenen Quellen stammen, hinsichtlich der Erläuterungen in englischer Sprache abgefaßt.

Beim Arbeiten mit fremden Programmen werden Sie deshalb sehr oft vor zwei Problemen stehen:

- Sie müssen Programm und Beiwerk verstehen und dazu – so weit notwendig – ins Deutsche übersetzen.
- Sie müssen prüfen, ob es auf Ihrem Computer läuft; ggf. müssen Sie es dazu ein wenig abändern.

Ich denke, wir sollten das an einem Beispiel einüben. Wir tun das an einem Programm, das ich einem der zahlreichen „Programmbücher" *) entnommen habe. Es handelt sich um ein Spielprogramm, das Ihren Computer in die Lage versetzt, gegen Sie ein Kartenspiel zu spielen; und zwar das bekannte Spiel „Siebzehn und Vier".

Damit Sie eine Vorstellung davon bekommen, wie derartige Programme aussehen können, folgt das ausgewählte Spielprogramm auf den nächsten Seiten im Original; d. h., so, wie es in dem erwähnten Buch steht.

```
10  REM BLACKJACK
20  PRINT "IF INSTRUCTIONS ARE REQUIRED TYPE YES"
30  PRINT "IF NOT TYPE NO"
40  INPUT C$
```

*) KEN TRACTON: „57 Practical Programs & Games in BASIC"
TAB Books Nr. 1000, TAB BOOKS Blue Ridge Summit Pa. 17214/USA

```
 50  IF C$ = "YES" THEN 90
 60  IF C$ = "NO" THEN 340
 70  PRINT "INVALID RESPONSE"
 80  GOTO 20
 90  PRINT
100  PRINT"==============BLACKJACK=================
= = = ===="
110  PRINT
120  PRINT "THE COMPUTER AS THE DEALER, DEALS
          TWO CARDS TO ITSELF"
130  PRINT "AND TWO CARDS TO THE PLAYER. THE PLAYER'S TWO
          CARDS"
140  PRINT "ARE SHOWN FACE UP, WHILE ONLY ONE OF THE DEALER'S"
150  PRINT "CARDS IS SHOWN. BOTH THE DEALER AND THE PLAYER"
160  PRINT "MAY DRAW ADDITIONAL CARDS."
170  PRINT "THE PLAYER'S GOAL IS TO REACH 21 OR LESS, BUT"
180  PRINT "BE CLOSER TO 21 THAN THE DEALER'S HAND."
190  PRINT "IF THE PLAYER'S OR THE DEALER'S HAND TOTALS"
200  PRINT "GREATER THAN 21 HE IS BUSTED! THE KING"
210  PRINT "THE QUEEN AND THE JACK ALL COUNT AS 10 POINTS."
220  PRINT "ALL OTHER CARDS EXCEPT THE ACE COUNT AS THEIR FACE"
230  PRINT "VALUE SHOWS. THE ACE COUNTS AS 11 UNLESS THIS"
240  PRINT "WOULD CAUSE THE HAND TO BE OVER 21, IN THAT"
250  PRINT "CASE THE ACE COUNTS AS 1."
260  PRINT "IF BOTH THE DEALER AND THE PLAYER GET BLACKJACK"
270  PRINT "WHICH IS A TWO CARD HAND TOTALING 21"
280  PRINT "NEITHER WINS, IT IS A PUSH"
290  PRINT "IF THE DEALER'S HAND IS BELOW OR EQUAL TO 16"
300  PRINT "HE MUST DRAW, AFTER 17 THE DEALER MUST STAND"
310  PRINT "TO RECEIVE A CARD YOU WANT A HIT–"
320  PRINT "TO STOP WHERE YOU ARE; YOU STAND–"
330  PRINT
340  PRINT "***GOOD LUCK***MAY THE BEST ONE WIN***"
350  REM 1.ST HAND
355  RANDOMIZE
360  LET D = 0
370  LET P = D
380  GOSUB 820
390  LET D1 = C
400  GOSUB 820
410  LET D2 = C
```

```
420  GOSUB 890
430  LET P1 = C
440  GOSUB 890
450  LET P2 = 3
460  PRINT
470  PRINT "THE DEALER HAS A";D1;"SHOWING"
480  PRINT "YOU HAVE A ";P1;" AND A ";P2
490  PRINT "YOUR TOTAL IS ";P1 + P2
500  LET D = D1 + D2
510  LET P = P1 + P2
520  IF P = 21 THEN 640
530  GOSUB 960
540  IF L = 1 THEN 690

550  IF D <= 16 THEN 740
560  PRINT "THE DEALER HAS ";D
570  PRINT "YOU HAVE ";P
580  IF P > D THEN 620
590  REM WIN OR LOSS STATEMENTS
600  PRINT "THE DEALER HAS WON!!!"
610  GOTO 1060
620  PRINT "YOU HAVE WON!!!"
630  GOTO 1060
640  PRINT "***YOU HAVE BLACKJACK***"
650  IF D = 21 THEN 670
660  GOTO 560
670  PRINT "THE DEALER ALSO HAS BLACKJACK,
          SORRY NO WINNER"
680  GOTO 1060
690  GOSUB 890
700  PRINT "YOUR CARD IS ";C
710  IF P > 21 THEN 600
730  GOTO 530
740  PRINT "THE DEALER HAS ";D
750  GOSUB 820
760  LET D = D + C
770  PRINT "THE DEALER DRAWS A ";C
780  PRINT "HIS TOTAL IS ";D
790  IF D > 21 THEN 620
800  IF D <= 16 THEN 750
810  GOTO 560
```

```
820  LET C = 1 + INT(11*RND)
830  IF C = 11 THEN 850
840  GOTO 880
850  IF D + C > 21 THEN 870
860  GOTO 880
870  LET C = 1
880  RETURN
890  LET C = 1 + INT(11*RND)
900  IF C = 11 THEN 920
910  GOTO 950
920  IF P + C > 21 THEN 940
930  GOTO 950
940  LET C = 1
950  RETURN
960  PRINT "DO YOU WANT A HIT.OR DO YOU STAND"
970  INPUT Q$
980  IF Q$ = "HIT" THEN 1020
990  IF Q$ = "STAND" THEN 1040
1000 PRINT "INVALID RESPONSE"
1010 GOTO 960
1020 LET L = 1
1030 GOTO 1050
1040 LET L = 0
1050 RETURN
1060 PRINT
1070 PRINT "DO YOU WISH TO PLAY AGAIN"
1080 PRINT "TYPE YES OR NO"
1090 INPUT L$
1100 IF L$ = "YES" THEN 1130
1110 PRINT "BLACKJACK SAYS GOOD-BYE"
1120 STOP
1130 PRINT
1140 GOTO 20
1150 END
```

Haben Sie das Programm einmal durchgesehen? Dann werden Sie bemerken, daß in den Programmzeilen 120 bis 340 die „Spielregeln" enthalten sind, nach denen Programm und Spiel aufgebaut sind.

Ich werde deshalb diesen Teil zunächst einmal ins Deutsche übersetzen; für den weiteren Verlauf ist es wichtig, daß Sie die Spielregeln genau kennen.

```
20 PRINT"WENN SIE DIE SPIELREGELN SEHEN WOLLEN
                ..................................... GEBEN SIE JA EIN!"
30 PRINT"WENN NICHT ..........................................
                ..................................... GEBEN SIE NEIN EIN!"
40 INPUT C$
50 IF C$="JA" THEN 90
60 IF C$="NEIN" THEN 360
70 PRINT"FALSCHEINGABE! JA ODER NEIN EINGEBEN!"
80 GO TO 20
90 CLS
100 PRINT"********** SIEBZEHN UND VIER**********"
110 PRINT
120 PRINT"IHR COMPUTER SPIELT DIE BANK!
          ER GIBT SICH SELBST ZWEI KARTEN"
125 PRINT
130 PRINT"SIE ERHALTEN EBENFALLS ZWEI KARTEN"
140 PRINT"DIE WERTE IHRER KARTEN WERDEN IHNEN GEZEIGT"
150 PRINT"VON DEN ZWEI KARTEN DES COMPUTERS SEHEN SIE NUR EINE"
160 PRINT"BEIDE SPIELER KOENNEN ZUSAETZLICHE KARTEN ZIEHEN"
170 PRINT"IHR ZIEL IST ES, 21 ODER WENIGER ZU ERREICHEN. ABER SIE
          SOLLTEN DICHTER BEI 21 SEIN ALS DIE BANK"
180 PRINT"DER SPIELER HAT VERLOREN, DESSEN AUGENZAHL UEBER 21
          IST"
```

An dieser Stelle ist mir eingefallen, daß mein Computer nur 14 (von 16) Zeilen Text auf einmal auf den Schirm bringt. Das Programm sollte also eine Weile anhalten können – damit man die Regeln in Ruhe lesen kann – und dann mit dem Rest des Textes fortfahren.

```
185 FOR X = 1 TO 1000 : NEXT
188 CLS
190 PRINT"KOENIG, DAME UND BUBE HABEN DEN WERT 10, ALLE ANDE-
          REN – MIT AUSNAHME DES ASS' – DEN AUFGEDRUCKTEN
          WERT"
200 PRINT"DAS ASS ZAEHLT 11 – ES SEI DENN, DER SPIELER ODER DIE
          BANK WUERDEN DADURCH UEBER 21 KOMMEN"
205 PRINT
210 PRINT"IN DIESEM FALLE ZAEHLT DAS ASS 1"
215 PRINT"FALLS DIE BANK 16 ODER WENIGER HAT, MUSS SIE EINE KARTE
          ZIEHEN. AB 17 DARF SIE NICHT MEHR ZIEHEN"
212 PRINT"FALLS SPIELER ODER BANK MIT ZWEI KARTEN 21 ERREICHEN,
          HABEN SIE 'BLACKJACK' – ES GEWINNT NIEMAND!"
```

```
230 PRINT"SIE KOENNEN NACH BELIEBEN EINE KARTE ZIEHEN ODER
          PASSEN"
240 PRINT
250 PRINT"**********VIEL GLUECK!**********"
```

Ab Zeile 350 beginnt nun das eigentliche Programm. Es hat die Aufgabe:

- Den beiden Spielern – also Ihnen und Ihrem Computer, der die „Bank" spielt – zunächst je 2 Karten zu geben.
- Das Programm bedient sich dazu zweier Zufallsgeneratoren; je eines für die Bank und den Spieler.
- Dann muß das Programm den zahlreichen, zunächst nicht recht verständlichen Regeln folgen.
- Nach den Regeln muß es: – Ihnen weitere Karten anbieten und „zuweisen".
  – Die Bank „ziehen" lassen.
  – Den Sieger ermitteln und „melden".

Damit Sie sich mit den Regeln vertraut machen können, setzen wir das Programm wie folgt fort:

```
350 FOR X = 1 TO 1000 : NEXT
360 GO TO 100
```

Haben Sie alles erfaßt? Dann stoppen Sie das Programm mit BREAK und fahren Sie fort!

Sie könnten nun ab Zeile 360 das Originalprogramm eingeben und dabei allenfalls die Texte in den PRINT...-Zeilen übersetzen. Ich habe das auch so gemacht. Aber ich bin nicht recht glücklich mit dem Programm geworden. Daß in Zeile 450 ein Druckfehler steckt (es muß wohl ...LET P2 = C heißen statt ...3), habe ich rasch gemerkt. Aber...
Immerhin bin ich durch mein „Unglück" in der Lage, Ihnen einige weitere Merkregeln mit auf Ihren Weg zum Programmierer zu geben.

- Auch das, was Sie „schwarz auf weiß" haben, muß nicht zum Erfolg führen. Das kann an dem Programm liegen, an Ihnen oder daran, daß das Programm sich mit Ihrem Computer „nicht verträgt".
- Ein fremdes Programm gedanklich nachzuvollziehen, ist nicht einfach, wenn es – wie in unserem Beispiel – nicht verbal oder sonstwie erklärt ist. Das mag daran liegen, daß jeder Programmierer die Aufgabe anders anpackt; bekanntlich „führen viele Wege nach Rom..."
  Wenn Sie „basteln" – beispielsweise Radios oder sonstige Elektronikschaltungen – werden Sie davon eine gute Vorstellung haben. Es ist oft einfacher, ein Gerät noch einmal von Grund auf neu zu bauen, als in einem nicht funktionierenden mit unzureichenden Mitteln oder unzureichendem Wissen einen Fehler zu suchen.

Kurzum – ich habe mich entschlossen, das Programm selbst zu Ende zu machen – wobei mir das vorhandene gute Dienste geleistet hat.

An dieser „Entwicklungsarbeit" möchte ich Sie jetzt teilhaben lassen; Sie werden dabei eine Menge lernen.

Um das Problem zu verstehen, habe ich mir zunächst die unten stehende Abbildung gezeichnet, in dem die wichtigsten Programm-„Pakete" in Kästchen aufgeschrieben sind (*Abb. 31*).

Sie haben erkannt, daß D die Karten für den „DEALER" (das ist die Bank) und P die Karten für den „Player" (das sind Sie) sind. Zu Beginn des Spieles sind D und P gleich Ø.

Da ich im folgenden meine Erläuterungen in das Programm einstreue, mache ich zu Beginn jeder Programmzeile, die Sie in Ihren Computer eingeben sollen, einen „Punkt", damit Sie nicht den Überblick verlieren. Es geht los!

- 36Ø D = Ø
- 37Ø P = Ø

Jetzt verzweigt das Programm viermal hintereinander in die Unterprogramme 1ØØØ...1Ø6Ø und 11ØØ...116Ø. Dort stehen die Zufallsgeneratoren, die die Karten ausgeben.

RND(11) ist deshalb gewählt, weil in diesem Spiel die Karten zwischen 1 und 11 liegen.

In den Unterprogrammen wird laut Spielregeln gleichzeitig geprüft:

- Ist die gezogene Karte 11? (Zeilen 1Ø1Ø und 111Ø)
- Ergibt die gezogene Zahl mit der/mit den schon vorhandenen mehr als 21? (Zeilen 1Ø3Ø und 113Ø)
- In diesem Fall wird C = 1 gesetzt. Für die erste Runde kann das noch nicht zutreffen!

- 38Ø GOSUB 1ØØØ
- 39Ø D1 = C
- 4ØØ GOSUB 1ØØØ
- 41Ø D2 = C
- 42Ø GOSUB 11ØØ
- 43Ø P1 = C
- 44Ø GOSUB 11ØØ
- 45Ø P2 = C

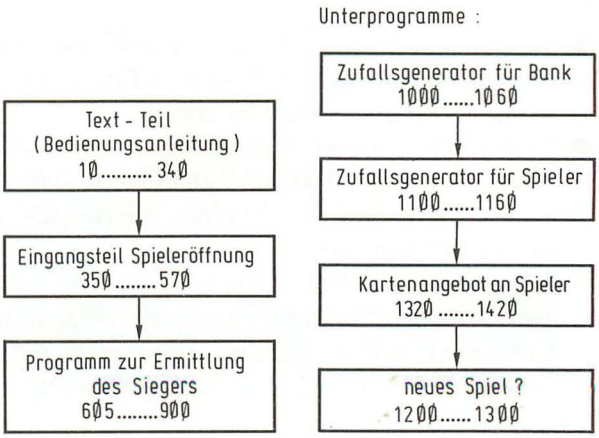


Abb. 31 Das Grob-Flußdiagramm zu dem Spiel „Black Jack"

Hier ist es erstmals von Interesse, das Programm auf richtige Funktion zu prüfen. Tun Sie das!

```
460 PRINT"KARTEN FUER DIE BANK"
470 PRINT"D1=";D1,,"D2=";D2
475 PRINT"KARTEN FUER DEN SPIELER"
480 PRINT"P1=";P1,,"P2=";P2
```

Lassen Sie das ein paarmal laufen, wobei Sie – um Zeit zu sparen – mit RUN 360 starten

Da bereits beim ersten Durchgang für Spieler und/oder Bank insgesamt 21 gezogen sein können, überprüfen wir das Ergebnis. Laut Spielregeln ist das der „Black-Jack"-Fall.

```
...IF P1 + P2 = 21 THEN...
...IF(P1+P2=21) AND (D1+D2=21) THEN...
```

wären angebrachte „Prüfmethoden"!

Treffen beide Prüfungen NICHT zu, dann gilt laut Spielregeln:

- Der Spieler kann beliebig eine weitere Karte nehmen oder ablehnen. Dazu benötigen wir ein „Karten-Angebotsprogramm".
- Die Bank MUSS eine weitere Karte ziehen, wenn die Summe der bisher gezogenen Karten 16 ODER KLEINER ist.
- Die Bank darf KEINE KARTE mehr ziehen, wenn die Summe der bisher gezogenen Karten 17 ODER GROESSER ist.
- Bevor der Spieler eine neue Karte angeboten bekommt, muß ihm sein bisher erreichtes Ergebnis „aufgedeckt" werden.
- Aber: von den beiden Karten, die für die Bank gezogen wurden, bekommt der Spieler NUR EINE zu sehen (damit das Spiel auch spannend bleibt!).

Machen wir weiter im Programm!

- 460 IF (P1+P2=21) AND (D1+D2=21) THEN 500
- 470 IF P1+P2=21 THEN 480
- 475 GO TO 530
- 480 PRINT "GRATULIERE! SIE HABEN 'BLACKJACK'!!"

Damit ist das erste Spiel bereits beendet und wir springen in einen Programmteil ab Zeile 1200, in dem ein NEUES SPIEL angeboten wird. Damit können wir uns im Augenblick noch Zeit lassen!

- 490 GO TO 1200
- 500 PRINT"SIE HABEN PECH! AUCH DIE BANK HAT 'BLACKJACK'!"
- 510 PRINT"ES GIBT DESHALB K E I N E N GEWINNER!!"
- 520 GO TO 1200

Jetzt müssen wir aber „die Karten auf den Tisch legen"!

- 53Ø CLS
- 535 PRINT"DIE BANK HAT E I N E KARTE MIT DEM WERT "; D1
- 54Ø PRINT"DIE ZWEITE KARTE DER BANK BLEIBT VERDECKT"
- 55Ø PRINT
- 56Ø PRINT"FUER SIE WURDEN ";P1;" UND ";P2;" GEZOGEN"
- 57Ø PRINT"SIE HABEN DAMIT BEREITS ";P1+P2
- 575 PRINT

Damit Sie das Ergebnis in Ruhe betrachten und danach Ihre Entscheidung für den weiteren Fortgang des Spiels machen können, folgt jetzt eine „Kunstpause".

- 58Ø FOR X = 1 TO 1ØØØ : NEXT

Jetzt müssen Sie in dem Programm:

- dem Spieler eine neue Karte anbieten.
- prüfen, ob die Bank „ziehen" muß oder nicht.

Wir fassen zunächst das bisher erreichte Ergebnis zusammen:

- 59Ø D = D1+D2
- 6ØØ P = P1+P2

Jetzt folgt das „Kartenangebot" an den Spieler, das in einem weiteren Unterprogramm ab Zeile 132Ø untergebracht ist.

- 61Ø GOSUB 132Ø

Und nun muß ich gestehen, daß ich bei der weiteren Programmentwicklung arg ins Schwitzen geraten bin. Ich will Ihnen die zahlreichen Irrwege nicht zumuten, sondern Ihnen nur die Probleme schildern, die mich so in „Streß" gebracht haben.

- Es ist im voraus nicht bekannt, wie oft der Spieler von dem Angebot, eine neue Karte zu ziehen, Gebrauch macht. Er muß ja darauf achten, daß er nicht über 21 kommt und darf das Spiel jederzeit abbrechen. Dieser Spielabbruch wird erreicht, wenn der Spieler *keine* neue Karte mehr haben will.
- Sie müssen in jeder neuen Runde nach den Spielregeln prüfen, ob auch die Bank eine neue Karte ziehen muß oder ob sie keine mehr ziehen darf.
- Sie müssen immer die Gesamt-Augenzahl der gezogenen Karten verfolgen.
- Hat der Spieler das Spiel durch Verweigerung einer neuen Karte abgebrochen, dann müssen Sie die *Gesamt-Augenzahl* beider Spieler ausgeben und
- entscheiden, wer gewonnen hat.

Das größte Problem war dabei zunächst, die Gesamt-Augenzahl zu errechnen.

Ich zeige Ihnen wieder an einer Graphik, wie ich am Ende den Rest des Programms aufgebaut habe (*Abb. 32*).

- 6Ø5 Y = Ø:Z = Ø

Das ist der Beginn von zwei Zählschleifen, mit denen die Gesamt-Augenzahlen beider Spieler ermittelt werden.

- 62Ø IF L = 1 THEN 64Ø

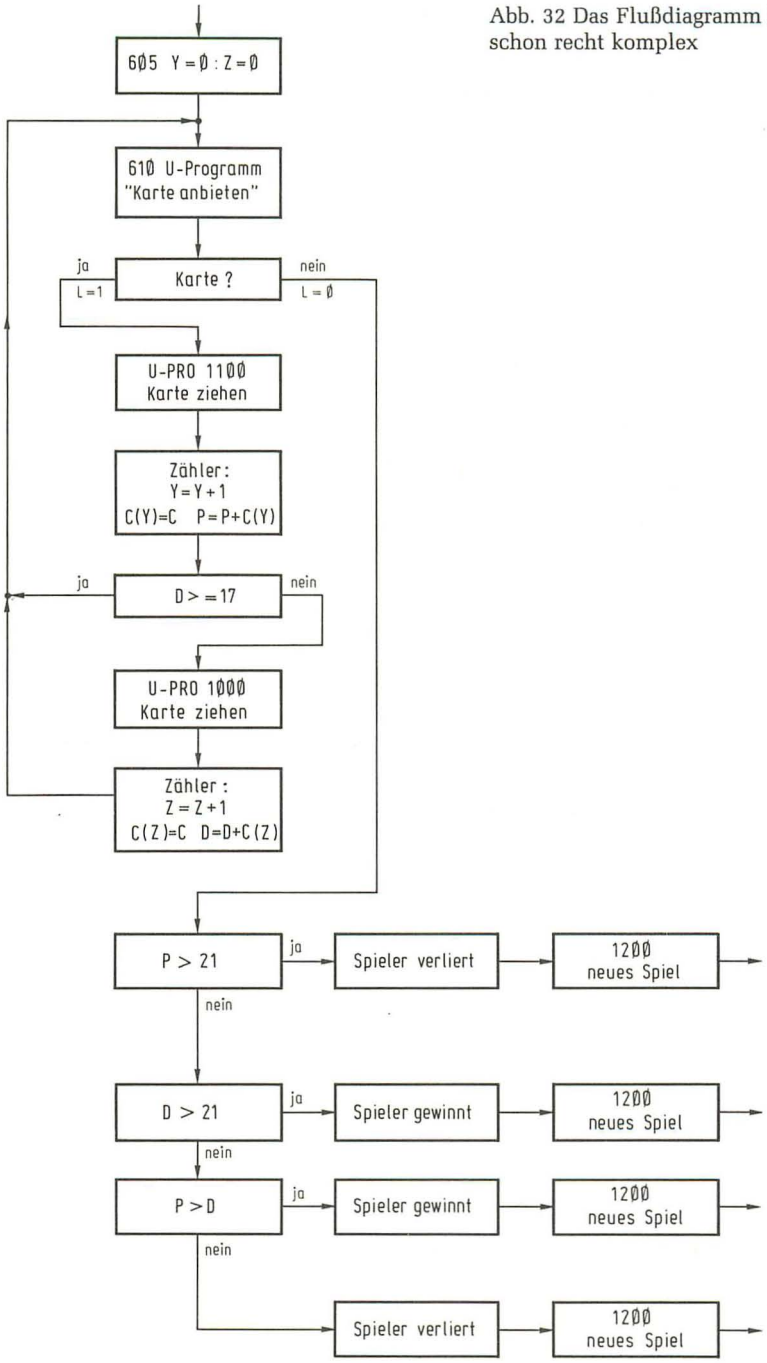


Abb. 32 Das Flußdiagramm ist schon recht komplex

MIT L = 1 gibt der Spieler im Unterprogramm ab 132Ø zu erkennen, daß er eine neue Karte haben möchte. Das bedeutet, daß der Computer zum Zufallsgenerator geht und eine neue Karte für den Spieler herbeischafft.

- 64Ø GOSUB 11ØØ
- 65Ø PRINT"SIE HABEN ";C;" GEZOGEN

Das kann Ihnen reichen – oder auch nicht. Um zu verfolgen, wie oft Sie ziehen, setzen wir den Zähler um eins weiter!

- 66Ø Y = Y+1

Jetzt wird's tückisch! Wir müssen die gezogene Karte aufaddieren. Und zwar mit einem Programm, das diese Aufgabe bewältigt, ohne zu wissen, wie oft gezogen wird!

Das können wir nicht machen mit den vom Computer gelieferten numerischen Variablen aus dem Zufallsgenerator (C). Um C „zählbar" zu machen, müssen wir die numerische in eine indizierte Variable umwandeln – weil wir bei denen die „Indices" mitzählen können! Das geschieht wie folgt:

- 67Ø C(Y) = C
- 68Ø P = P + C(Y)

Und nun sagen wir dem Spieler:

- 69Ø PRINT"SIE HABEN NUN ";P

Fairerweise müssen wir dem Spieler für seine nun fällige Entscheidung, ob er eine weitere Karte ziehen soll oder nicht, verraten, was die Bank gezogen hat. Dazu müssen wir die Bank erst mal ziehen lassen – nicht ohne vorher geprüft zu haben, ob die Bank auch ziehen darf!

- 7ØØ IF D >= 17 THEN 61Ø
- 71Ø GOSUB 1ØØØ
- 72Ø PRINT"DIE BANK HAT ";C;" GEZOGEN"

Mit den folgenden Programmzeilen verfolgen wir, was die Bank gezogen hat.

- 73Ø Z = Z + 1
- 74Ø C(Z) = C
- 75Ø D = D + C(Z)
- 77Ø GO TO 61Ø

Jetzt wollen wir annehmen, daß der Spieler im Unterprogramm ab 132Ø keine neue Karte gezogen hat. Er hat das durch die Eingabe von L = Ø zu erkennen gegeben. Sie müssen jetzt:

– das Spiel abbrechen
– ermitteln, wer der Sieger ist.
– In 62Ø war geprüft worden „...will der Spieler eine Karte...?" – und zwar durch

...IF L = 1...

– Will er keine, dann ist L NICHT 1 und es geht im Programm weiter:

- 63Ø GO TO 82Ø

Jetzt folgt die Prüfung: „Wer hat gewonnen?" – denn das Spiel ist ja beendet und Sie müssen das Ergebnis ermitteln und verkünden.

- 82Ø IF P > 21 THEN 845
- 83Ø IF D > 21 THEN 875
- 84Ø IF P > D THEN 875

Und nun verkünden Sie das Ergebnis des Spieles:

- 845 CLS
- 85Ø PRINT" ========== SIE HABEN V E R L O R E N ========="
- 855 PRINT
- 86Ø PRINT" SIE HABEN ";P;" DIE BANK HAT ";D
- 87Ø GO TO 12ØØ
- 875 CLS
- 88Ø PRINT" ========== SIE HABEN G E W O N N E N ======="
- 885 PRINT
- 89Ø PRINT"DIE BANK HAT ";D; SIE HABEN ";P
- 9ØØ GO TO 12ØØ

Es folgen jetzt noch die Unterprogramme für die Zufallsgeneratoren, für „Neues Spiel?" sowie der Programmblock „Kartenangebot".

```
1ØØØ C = RND(11)
1Ø1Ø IF C = 11 THEN 1Ø3Ø
1Ø2Ø GO TO 1Ø6Ø
1Ø3Ø IF D + C > 21 THEN 1Ø5Ø
1Ø4Ø GO TO 1Ø6Ø
1Ø5Ø C = 1
1Ø6Ø RETURN
```

ZUFALLSZAHL FÜR DIE BANK

```
11ØØ C = RND(11)
111Ø IF C = 11 THEN 113Ø
112Ø GO TO 116Ø
113Ø IF P + C > 21 THEN 115Ø
114Ø GO TO 116Ø
115Ø C = 1
116Ø RETURN
```

ZUFALLSZAHL FÜR DEN SPIELER

```
1320 PRINT"WUENSCHEN SIE NOCH EINE KARTE?"
1330 PRINT„TIPPEN SIE 1 FUER JA, 0 FUER NEIN"
1340 INPUT B
1350 IF B = 1 THEN 1390
1360 IF B = 0 THEN 1410
1370 PRINT"UNGUELTIGE EINGABE!"
1380 GO TO 1420
1390 L = 1
1400 GO TO 1420
1410 L = 0
1420 RETURN
```

"NEUE KARTE?"

```
1200 FOR X = 1 TO 1000 : NEXT
1205 CLS
1210 PRINT"WUENSCHEN SIE EIN NEUES SPIEL?"
1220 PRINT"TIPPEN SIE 1 FUER JA, 0 FUER NEIN"
1230 INPUT A
1240 IF A = 1 THEN 1280
1250 IF A = 0 THEN 1290
1260 PRINT"FALSCHEINGABE!"
1270 GO TO 1200
1280 GO TO 250
1290 PRINT"SCHADE! AUF WIEDERSEHEN!
            SIE WAREN EIN PRIMA GEGNER!"
1300 END
```

"NEUES SPIEL?"

39.4 Sie erweitern den Tabulator Ihres Computers

In vielen Programmen haben Sie die Fähigkeit Ihres Computers genutzt, auf ein Komma hin eine Spalte weiter zu rücken. Wie die meisten Personal-Computer hat der TRS-80 die Fähigkeit, mit Kommas max. vier Spalten auf den Bildschirm zu bringen.

Es gibt in Basic einen Befehl, mit dem die „Tabellierfähigkeit" des Computers erweitert werden kann. Dieser Befehl heißt: TAB (...), wobei TAB für TABulator steht.

Üben Sie das in einem kleinen Programm!

```
NEW
5 CLS
10 PRINT TAB(10) "W"
20 PRINT TAB(11) "E"
```

```
30 PRINT TAB(12) "B"
40 PRINT TAB(13) "E"
50 PRINT TAB(14) "R"
```

Dieses Programm führt zu folgendem Ergebnis:

```
        Spalte 10
        ↓
-------------------------------
        W
          E
            B
              E
                R
```

Sie haben also in (...) befohlen, wo der Tabulator die neue Zeile beginnen soll; im Beispiel hat er den ersten Buchstaben (W) in die 10. Spalte gesetzt und ist danach jeweils um eine Zeichenposition weitergerückt.
Das gleiche Ergebnis erhalten Sie mit:

```
10 PRINT TAB(10)"W":PRINT TAB(11)"E":PRINT TAB(12)"B": PRINT TAB
       (13)"E":PRINT TAB(14)"R"
```

Bitte merken Sie sich:

- Der TAB(...)-Befehl funktioniert nur zusammen mit dem PRINT-Befehl.

Versuchen Sie sich jetzt einmal an einem sinnvolleren Gebrauch des TAB(...)-Befehls!

```
NEW
5 CLS
10 X = 1
20 PRINT TAB(2) X*2;TAB(8) X*3;TAB(14) X*4;TAB(20) X*5;TAB(26) X*6;
   TAB(32) X*7;TAB(38) X*8;TAB(42) X*9;TAB(50) X*10
30 X = X + 1
40 IF X > 10 THEN 60
50 GO TO 20
60 END
```

Ist das Ergebnis nicht eine gelungene „Rechentafel"?

Bitte merken Sie sich:

- Bei einer Darstellungsart wie der oben gezeigten brauchen Sie nur zu Anfang den PRINT-Befehl.
- Sie müssen die einzelnen Abschnitte mit ; (Semikolon) voneinander trennen, damit der Computer weiß, daß er weitermachen und nicht zu einer neuen Zeile gehen soll.

- Sie müssen beachten, wieviele Zeichen maximal auf eine Zeile passen. Beim TRS-80 sind das – wie bei den meisten Personal-Computern – 64 Zeichen.

Wenn Sie Lust haben, können Sie ja mal alle Programme daraufhin durchsehen, ob sich nicht der „Informationsinhalt" der Computerausgabe durch die Anwendung des TAB(...) Befehls erweitern läßt!

39.5 Noch einmal: Lottozahlen!

Kramen wir noch einmal folgendes Programm hervor:

```
NEW
5 CLS
10 FOR N = 1 TO 6
20 T(N) = RND(49)
30 PRINT T(N)
40 NEXT N
```

Sie haben den Mechanismus des Zufallsgenerators noch im Kopf? Dann wissen Sie, daß dieses Programm innerhalb der 6 ausgegebenen Zahlen zwar alle zwischen 1 und 49 anordnet, daß aber in einer Zufallszahlenreihe auch zweimal die gleiche Zahl erscheinen kann. Mir ist das zwar erst beim 12. Durchlauf passiert. Aber es kann genau so gut beim 2. oder beim 35. geschehen!

Wir sollten das Programm so erweitern, daß es jede gezogene Zahl daraufhin überprüft, ob sie schon einmal gezogen wurde. Falls das zutrifft, sollte diese Zahl verworfen und eine neue gezogen werden; solange, bis keine Dopplungen mehr auftreten.

Fangen Sie noch einmal von vorn an!

Sie erledigen das „Prüfen" und „Verwerfen" mit zwei ineinander verschachtelten FOR...-NEXT-Schleifen.

```
30 IF N = 1 THEN 70
40 FOR M = 1 TO N - 1
50 IF T(N) = T(M) THEN 20
60 NEXT M
70 NEXT N
```

Nun lassen Sie sich die Liste ausgeben!

```
80 FOR N = 1 TO 6
90 PRINT T(N)
100 NEXT N
```

Lassen Sie das Programm so oft „fahren", bis Sie mir glauben, daß keine Zahl zweimal kommt!

Sie haben bemerkt, daß die erwähnte Prüfung in den Zeilen 4Ø bis 6Ø geschieht? Ich hatte beim ersten Durchlauf des Programms folgendes Ergebnis:

```
42
27
 3
12
44
 7
```

Daran stört mich, daß die Zahlen genauso durcheinander stehen, wie sie der Computer gezogen hat. Wozu haben wir das Sortieren gelernt?

Die nächsten Programmzeilen sortieren jetzt die vom Computer gezogenen Zahlen in aufsteigender Reihenfolge. Dabei führe ich Ihnen gleichzeitig ein neues, recht einfaches Sortierverfahren für Zahlen vor. Dieses Sortierprogramm hat eine gewisse Ähnlichkeit mit den bereits besprochenen. Der deutlichste Unterschied steckt in der Vergleichszeile; und zwar wird nicht geprüft, ob die erste Zahl kleiner ist als die zweite, sondern ob sie größer ist!

Lassen Sie sich überraschen!

... A = 5 hier stand früher: A = N – 1, wobei N die Anzahl der zu sortierenden Zahlen war. Da Sie hier 6 Zahlen sortieren wollen, können Sie es sich einfacher machen!

```
8Ø S = Ø
9Ø FOR N = 1 TO 5
1ØØ IF T(N+1) > = T(N) THEN 16Ø
11Ø H = T(N)
12Ø T(N) = T(N+1)
13Ø T(N+1)=H
14Ø S = 1
16Ø NEXT N
165 IF S > Ø THEN 8Ø
```

Lassen Sie sich den „Tip der Woche" Ihres Computers ausgeben!

```
17Ø PRINT"MEIN TIP DER WOCHE LAUTET:"
18Ø PRINT" ==========================================
======"
19Ø FOR N = 1 TO 6
2ØØ PRINT T(N)
21Ø NEXT N
```

Ich bemerke schon, daß Sie auf dem besten Wege zum Perfektionisten sind! Ihnen mißfällt, daß das Ergebnis untereinander steht!
Dann setzen Sie es halt nebeneinander!

```
190 PRINT TAB(2) T(1);TAB(8) T(2); TAB(14) T(3); TAB(20) T(4);
          TAB(26) T(5); TAB(32) T(6); TAB(36) T(6)
200
210
```

Zum Lotto-Spielen ist mir noch etwas eingefallen! Das Ergebnis der Zahlenziehung erleben Sie im Fernsehen ja so, daß nicht nur die sechs Richtigen angezeigt werden, sondern auch noch die Zusatzzahl.

Diese wird beim Tippen bekanntlich nicht angekreuzt; wenn wir also noch etwas mit der Zusatzzahl anfangen wollen, dann müssen wir unser Programm umwandeln zu einem „Lotto-Vorhersage-Programm"!

Das nächste Programm soll also alle sechs Lottozahlen in aufsteigender Reihenfolge ausgeben und zum Schluß die Zusatzzahl vorhersagen.

Einfach, werden Sie denken! Ich lasse einfach sieben Zahlen ziehen und ändere Zeile 10 zu: ...FOR N = 1 TO 7.

Machen Sie das doch mal! Die Änderung in Zeile 10 hat zwar noch Auswirkungen auf einige andere Programmzeilen. Aber das werden Sie selbst herausfinden! Lassen Sie das ein paarmal laufen. Fällt Ihnen etwas auf? Es sollte Ihnen auffallen, daß jetzt die Zusatzzahl – also die siebte gezogene Zahl – immer die größte Zahl ist! Und das stimmt überhaupt nicht mit der Prozedur des Lottozahlen-„Ziehens" überein! Bei der sonnabendlichen Ziehung wird zwar die Zusatzzahl als letzte gezogen. Aber es kann jede der 49 – 6 verbleibenden Zahlen sein; Beispielsweise auch 1 oder 2!

Ein neues Programm muß her!

Wir gehen von dem letzten Programm aus – wenn Sie es schon gelöscht haben, müssen Sie leider noch einmal von vorn anfangen!

Dieses Programm ergänzen wir um:

```
10 FOR N = 1 TO 7
75 Z = T(7)
```

Mit dem Befehl in Zeile 75 suchen wir also die siebte gezogene Zahl T(7) aus den Zufallszahlen heraus und machen sie zu Z. Der Rest des Programmes bleibt wie gehabt; insbesondere sortieren wir jetzt nur noch die ersten sechs gezogenen Zahlen. Nämlich durch:

```
90 FOR N = 1 TO 5 usw.!
```

Jetzt fügen Sie nur noch Ihrem letzten Programm an:

```
200 PRINT"DIE ZUSATZZAHL LAUTET: "; Z
```

Dieses hier gezeigte „Auf die Seite bringen" (von T(7)) können Sie bestimmt noch in anderen Programmen gebrauchen!

Ich hoffe, Sie gewinnen nächste Woche im Lotto!

39.6 Sind Sie ein Sonntagskind?

Sie wissen das nicht? Kein Problem; mit dem folgenden Programm wird der Computer Ihnen darüber Gewißheit verschaffen.

```
 10 PRINT"PROGRAMM ZUR ERMITTLUNG DES WOCHENTAGES ZU EINEM
          BESTIMMTEN DATUM"
 20 PRINT" ==========================================
======"
 30 J$(1) ="SONNTAG"
 40 J$(2) ="MONTAG"
 50 J$(3) ="DIENSTAG"
 60 J$(4) ="MITTWOCH"
 70 J$(5) ="DONNERSTAG"
 80 J$(6) ="FREITAG"
 90 J$(7) ="SONNABEND"
100 PRINT"GEBEN SIE DAS DATUM, ZU DEM SIE DEN WOCHENTAG
          SUCHEN, AUF ANFORDERUNG EIN! ZB ALS 22 12 1980"
110 INPUT"TAG";T
120 INPUT"MONAT";M
130 INPUT"JAHR";J
150 K = INT(.6+(1/M))
160 L = J - K
170 O = M + 12 * K
180 P = L/100
190 Z1 = INT(P/4)
200 Z2 = INT(P)
210 Z3 = INT((5*L)/4)
220 Z4 = INT(13*(O+1)/5)
230 Z = Z4+Z3-Z2+Z1+T-1
240 Z = (Z-(7*INT(Z/7)))+1
250 PRINT"DER GESUCHTE WOCHENTAG IST EIN ";J$(Z)
260 PRINT" ========================================="
```

In den Zeilen 150 bis 240 wird Z für J$(Z) und damit für den gesuchten Wochentag berechnet. Wie das zustande kommt, weiß ich beim besten Willen nicht. Ich habe dieses Programm in dem schon erwähnten Programmbuch gefunden und für Sie abgewandelt.

Es funktionierte auf Anhieb vorzüglich. Es gilt mit einer Einschränkung: Das eingegebene Datum muß nach dem Jahre 1753 liegen. Weil damals der Julianische Kalender durch den Gregorianischen ersetzt wurde.

Anhang

Spezielle Tastenfunktionen

In diesem Buch werden die Programmbeispiele mit dem TRS-80 erläutert. Soweit dazu Tasten mit besonderer Beschriftung verwendet werden, enthält diese Liste eine Übersicht über die Tasten für gleichlautende Funktionen für bekannte Computer anderer Fabrikate. In den ersten Kapiteln sind neben den Tastensymbolen für den TRS-80 Leerfelder gezeichnet, in die das entsprechende Symbol für andere Computer eingetragen werden kann:

Kennziffer	TRS-80	Auswirkung der Taste(n)	Apple/ ITT 2020	PET/CBM...	PC 100
①	ENTER	Abbruch der laufenden, Beginn einer neuen Zeile	RETURN	RETURN	RETURN
②	←	Cursor eine Zeichenposition nach links unter Löschen des dort stehenden Zeichens	← Bei Erreichen der gewünschten Position: *Leertaste drücken*	INST DEL	DEL
③	SHIFT ←	Löschen der gesamten, aktuellen Zeile	fehlt	fehlt	SHIFT @

④	* :	Doppelpunkt trennt BASIC-Befehle in einer Programmzeile voneinander	* :	* :	* :
⑤	CLEAR	löscht den Bildschirm, behält dabei das eingegebene Programm	zuerst: ESC danach: SHIFT @	SHIFT CLR+ HOME	fehlt
⑥	SHIFT @ weiter mit beliebiger Taste	Stoppt Programmausführung	CTRL C	RUN STOP	F 1 Neustart mit RUN
⑦	BREAK weiter mit CONT	Stoppt Programmausführung	CTRL bei ...2020 RESET RESET bei Apple	RUN STOP weiter mit CONT	F 1 Neustart mit RUN
⑧	↑	Mathematisches Zeichen für „hoch"; Beispiel 2^3 2 ↑. 3	keine spezielle Taste. Funktion per Programm eingeben (siehe Handbuch).	↑	F 3
⑨	SHIFT ? /	Abkürzung für PRINT	SHIFT ? /	SHIFT ? /	SHIFT ? /

Liste der in diesem Buch verwendeten Basic-Befehle

Im folgenden werden noch einmal alle in diesem Buch verwendeten Basic-Befehle ausgeführt. Neben einer kurzen Erklärung, in welchem Sinne die einzelnen Befehle verwendet wurden, folgt ein Hinweis auf diejenige Seite, von der ab der Befehl an Beispielen vorgeführt wird.

Befehl	Bedeutung in diesem Buch	erklärt ab Seite
AND	wird verwendet für logische Entscheidungen ...IF A=1 AND B=1 THEN... Kurzform: IF(A=1)*(B=1)THEN...	174
CLEAR	a) setzt alle numerischen und Stringvariablen zu „Null"; b) reserviert eine Anzahl von Bytes im Speicher für Stringvariable. Beispiel: CLEAR 1ØØØ	24
CLS	CLEAR SCREEN löscht den Bildschirm, ohne ein bestehendes Programm zu beeinflussen.	24
DIM	DIMensioniert bestimmte Speicherplätze für Variable. Beispiel: DIM SB(5Ø)	150
END	beendet an dieser Stelle die Abarbeitung eines Programms.	48
FOR...NEXT	wird verwendet für wiederkehrende Operationen. Beispiel: FOR X = 1 TO 1ØØØ „für X von 1 bis 1ØØØ"... FOR... muß mit ...NEXT abgeschlossen werden.	50
GOSUB...	Aufforderung, in ein Unterprogramm zu springen. Bei GOSUB 1ØØØ muß das Unterprogramm in Zeile 1ØØØ beginnen. Nach Abarbeiten des Unterprogramms muß mit RETURN ins Hauptprogramm zurückgesprungen werden.	117
GO TO...	Aufforderung, an eine bestimmte Zeile zu springen. Bei 1ØØ GO TO... kann das Ziel vor oder nach 1ØØ liegen.	62
IF...THEN	Ist der auf IF... folgende Ausdruck wahr, folgt Ausführung des nach ...THEN Folgenden. Beispiel: IF X = 1ØØ THEN 15Ø Wenn X = 100 ist, wird das Programm in 15Ø fortgesetzt.	64
INPUT	Eingabebefehl, der Dialoge mit dem Computer erlaubt. Auf INPUT A$ wartet der Computer mit der weiteren Abarbeitung des Programms, bis für A$ ein String eingegeben wird.	60
INT(...)	INTeger heißt „ganze Zahl". Auf INT(6.5) gibt der Computer den Ganzzahl-Anteil – hier 6 – aus.	95
LET D =	Zuweisungsbefehl. Mit LET... wird einer numerischen	46

D =	oder Stringvariablen ein bestimmter Wert zugewiesen. Beispiel: LET D = 1Ø weist D den Wert 1Ø zu Bei den meisten Computern kann LET... weggelassen werden; die Kurzform lautet dann: D = 1Ø	46
LIST	Befehl zum AufLISTen des Programms.	44
NEW	löscht ein im Speicher vorhandenes Programm.	44
ON...GO TO	Variante des Befehls ...GO TO... ...INPUT Q ...ON Q GO TO 1ØØ,15Ø,25Ø Je nach Eingabe für Q wird das Programm in 1ØØ oder 15Ø oder 25Ø fortgesetzt.	69
OR	wird verwendet für logische Entscheidungen ...IF A=1 OR B=1 THEN... Kurzform: IF(A=1)+(B=1)THEN...	173
PRINT	Ausführungsbefehl. In PRINT 3*5 versteht der Computer PRINT als Rechenbefehl und gibt aus: 15 Auf PRINT"3*5" versteht der Computer PRINT"..." als Druckbefehl und gibt aus: 3*5 Eine Kurzform für PRINT ist? Also ?3*5 statt PRINT3*5	26
PRINT"..."	Siehe oben!	30
PRINT TAB(...)	TAB ist ein Tabellierbefehl. Bei PRINT TAB(6) A „druckt" der Computer A an die 6. Stelle der Zeile; er rückt A 5 Stellen nach rechts!	218
READ...DATA	Dieser Befehl bedeutet: „lese ‚(READ)' die in einer Datenliste ‚(DATA)' abgelegten Daten." Beispiel: ...READ A ...PRINT A ...DATA 1,2,3	105
REM	ist die Abkürzung für REMark. Mit REM kann man in ein Computerprogramm Kommentare schreiben, die das Programm verständlich machen. Alles, was in einer Zeile auf REM folgt, wird vom Computer bei der Ausführung des Programms ignoriert.	78
RESTORE	veranlaßt den Computer, nach READ...DATA und Abarbeiten des folgenden Befehls in die DATA-Liste zurückzukehren und die gleichen Daten noch einmal (für einen anderen Befehl) zu verwenden.	113
RETURN	bringt den Computer aus einem Unterprogramm (...GOSUB...) zurück in das Hauptprogramm.	117
RND(...)	RND ist Abkürzung für „Random" – das heißt Zufall. Auf den Befehl RND(...) reagieren die einzelnen Computer wie folgt: Beispiel: Es soll eine Zufallszahl zwischen 1 und 49 gezogen werden. Es ist einzugeben:	120

	a) beim TRS-8Ø ...RND(49) b) beim PC 1ØØ Apple/ITT2Ø2Ø PET/CBM...:...INT(49*RND(1)+1)	
RUN	RUN ist ein Ausführungsbefehl. Soll ein eingegebenes Programm gestartet werden, so ist RUN einzugeben, danach RETURN bzw. ENTER.	40
RUN 5ØØ	Wie oben; hier wird das Programm jedoch erst ab Programmzeile 5ØØ abgearbeitet.	96
...STEP...	Bestandteil der FOR...NEXT-Schleife. STEP gibt vor, in welcher Schrittweise die FOR...NEXT-Schleife arbeiten soll. Beispiel: ...FOR X = 1 TO 1Ø STEP .1 heißt: „für x von 1 bis 1Ø in der Schrittweite .1"	52
SQR(...)	Zeichen für „Sqareroot"= Quadratwurzel	36
TAB(...)	Tabellierbefehl. Die Zahl in (...) gibt an, um wieviele Stellen nach rechts gerückt wird. TAB(5) rückt um 5 Stellen nach rechts, das erste Zeichen erscheint dann auf der 6. Position. TAB(...) arbeitet nur zusammen mit PRINT.	218
VAL(...)	VAL ist die Abkürzung für VALue = „Wert". Der nach VAL in (...) stehende Ausdruck muß ein String sein, der – für sich betrachtet – als Zahl einen Sinn ergibt. Beispiel: ...X$="125" ...PRINT VAL(X$) ergibt den Ausdruck: 125 ...X$="A125" ist VERBOTEN, weil A125 keine Zahl ist!	168

Die Bedeutung einiger Zeichen in Basic

. (Punkt)	wird in BASIC anstelle von , (Komma) verwendet. Beispiel: 3.5 ist richtig: „drei-Komma-fünf" 3,5 ist falsch
, (Komma)	Tabellierzeichen. Mit , können meist vier Spalten (beim PC 100 zwei Spalten) geschrieben werden. Ein , trennt ferner Daten in DATA...-Zeilen.
; (Strichpunkt)	veranlaßt in einer Programmzeile mit PRINT die unmittelbare Ausführung des nach ; Stehenden. Beispiel ...A=5 ...B=1Ø ...C=A*B ...PRINT"A*B=";C Ausdruck: A*B=5Ø

"..."	Anführungszeichen. Strings müssen in "..." gesetzt werden.
+	a) Zeichen für „Plus" (Addition) b) Kurzzeichen für logisches „Oder" (OR)
–	Zeichen für„Minus" (Subtraktion)
*	a) Zeichen für „Mal" (Multiplikation) b) Kurzzeichen für logisches „Und" (AND)
/	Zeichen für „Geteilt durch" (Division)
:	Doppelpunkt. Trennt Basic-Befehle innerhalb einer Programmzeile voneinander. Beispiel: ...FOR X = 1 TO 1ØØØ : NEXT

Zeichen zum Test auf Relationen

=	Gleichheitszeichen. ...A=5
<>	Zeichen für „ungleich"; „nicht gleich". ...A<>B
>	Zeichen für „Größer als". ...7>5
<	Zeichen für „kleiner als". ...5<7
>= bzw. =>	Zeichen für „kleiner oder gleich".

Die Variablentypen

A...Z A1...A9	Numerische Variable Beispiele: A=123; A9=12.3
A$...Z$ A1$...A9$	Stringvariable Beispiele: A$="EMIL"; A9$="E6Ø5"
A(1) S(2,4) A$(I)	Indizierte Variable

Literatur

Weber, W.: Einführung in die Methoden der Digitaltechnik. Allgemeine Elektrizitäts-Gesellschaft, Berlin

VDI: Programmierte Unterweisung. VDI-Verlag, Düsseldorf

Pütz, Jean: Digitaltechnik, eine Einführung. VDI-Verlag, Düsseldorf

Wolters, M. F.: Der Schlüssel zum Computer, Econ-Verlag, Düsseldorf

Mikroprozessoren und Mikrocomputer. Siemens AG, München

Level II Basic Reference Manual. Tandy Corp., Fort Worth, USA

Basic-Handbuch Personal Computer PC-100. Siemens AG, München

Abeldt, G.: Basic – Grundlagen und Beispiele. Frech Verlag, Stuttgart

Schärf, J.: Basic für Anfänger. R. Oldenbourg Verlag, München

Schärf, J., Kunesch, A.: Basic für Kaufleute, R. Oldenbourg Verlag, München

Ramp, H. O.: Basic-Praxis. R. Oldenbourg Verlag, München

Rehbein, H.: Basic leicht gemacht. VDI-Verlag, Düsseldorf

Kaufmann, K. D., Kizan, P.: Spaß mit Basic. Idea-Verlag, Puchheim

Wittig, S.: Basic-Brevier. Verlag Heinz Heise, Hannover

Feichtinger, H.: Basic für Mikrocomputer, Franzis-Verlag, München

Lien, D. A.: The Basic Handbook. Compusoft Publishing, San Diego, USA

Tracton, K.: 57 Practical Programms & Games in Basic. TAB Books, Blue Ridge Summit Pb.17214, USA

Friedrich: Tabellenbuch für Elektrotechnik. Ferd. Dümmler Verlag, Bonn

Stichwortverzeichnis

Beim Umgang mit Computern stoßen Sie oft auf spezielle Begriffe, deren Bedeutung in dem folgenden Verzeichnis näher erläutert wird. Soweit diese aus dem englischen Sprachraum stammen, sind sie sinngemäß übersetzt.

Stichwort	Übersetzung	Erläuterung
A		
Access	Zugriff	Möglichkeit, eine bestimmte Speicherstelle anzusprechen (zu adressieren) und zu lesen.
Adresse		Bezeichnungscode für eine Speicherstelle.
AD-Converter	AD-Wandler	Analog-Digital-Wandler: Schaltung, die analoge Werte in Digitalcode ausdrückt.
Accumulator	Akkumulator	Register mit der zusätzlichen Möglichkeit der binären Addition.
Algol		Algorithmic Programming Language: höhere Programmiersprache für technisch-wissenschaftliche Programme.
ALU	Rechenwerk	(Arithmetic und Logical Unit) Teil einer Computer-Zentraleinheit, in der arithmetische und logische Operationen vorgenommen werden.
Anwender-programm		Spezialprogramm, das ein Anwenderproblem behandelt (im Gegensatz zu Universal- oder Betriebsprogrammen).
Arbeitsprogramm		Programm, das datenverarbeitende Probleme in engem Sinn ablöst (also kein Übersetzungs-, Hilfs- oder Betriebsprogramm).
Arbeitsspeicher		relativ schneller Speicher für Programm und Zwischenergebnisse. Gegensatz: Massenspeicher.
ASCII		alphanumerischer Code (American Standard Code for Information Interchange).
Assembler		maschinennahe und maschinenspezifische Programmsprache, niedriges Niveau. Gleichzeitig:
Assembler		Übersetzungsprogramm, das ein in Assembler geschriebenes Programm in Maschinencode übersetzt.
asynchron		ohne Takt, taktunabhängig

B

Basic		einfach zu erlernende höhere Computerprogrammier-Sprache (Beginners All purpose Symbolic Instruction Code).
BCD-Ziffer		(Binary Coded Decimal) in 4 bit (Binärcode) codierte Ziffer.
Befehl	Instruction	Anweisung an den Rechner zur Ausführung einer Aktion.
Befehlsvorrat		Gesamtheit der Instruktionen, die ein Computer „versteht".
Befehlszähler	Program Counter	Register, in dem die Speicheradresse des nächsten zu bearbeitenden Befehls steht.
Betriebssystem	Operating System	Programmpaket, das die Bearbeitung von Programmen durch den Computer bequem möglich macht.
binäre Variable		Variable, die zwei Zustände annehmen kann (wahr –falsch bzw. JA–NEIN bzw. HIGH–LOW bzw. 1–Ø).
Binärzähler		Halbleiterbaustein, meist bestehend aus vier Zählflipflops, der dann von Ø bis 15 zählt.
Bit		(Binary Digit) binäre Informationseinheit
Boolsche Algebra		System von Rechenregeln für binäre Variable („Logische Operationen").
Bus		Datenleitung, an der mehrere Einheiten gleichzeitig angeschlossen sind (Adreßbus, Datenbus, Kontrollbus).
Byte		Kleinste adressierbare Einheit in Großanlagen (8 bit + 1 Parity-bit).

C

Chip		bezeichnet ein nacktes, fertig diffundiertes Siliziumplättchen oder aber auch einen Schaltkreis.
Clock	Takt	praktisch alle existierenden Computer sind „Synchron-Maschinen", d. h. ihre Operationen erfolgen in einem festgelegten, von einem Taktgenerator bestimmten Zeitraster.
CMOS		Complementary MOS: MOS-Technologie mit vernachlässigbarem Ruhestrom, mittelschnell.
COBOL		höhere Programmsprache für kommerzielle DV (Common Business Oriented Language).
Codieren		speziell in der Programmiertechnik: Umsetzen eines Programmablaufs in die Programmsprache.
(to) compute	rechnen	dieses englische Wort ist die Wurzel des Namens COMPUTER.
Computer		beliebig programmierbare Logikeinheit.

CPU	Zentraleinheit	Central Processing Unit: Rechenwerk und Steuerwerk eines Computers.
Cursor	Zeiger	gibt auf dem Bildschirm des (Daten-)Sichtgerätes an, wo das nächste eingegebene Zeichen erscheint.

D

Daten	Data	allgemein: in irgendeiner Form dargestellte Sachverhalte oder Vorgänge. Speziell versteht man darunter meist Ein- und Ausgabeinformation eines Computersystems. Es gibt numerische (Ziffern, Zahlen) und alphanumerische (Ziffern, Buchstaben, Sonderzeichen).
Datenbus		gemeinsame Datenschiene (-leitung) für mehrere gleichzeitig angeschlossene Einheiten.
Datenflußplan		Hilfsmittel beim DV-Systemaufbau. Aus dem Datenflußplan gehen Ein-, Ausgabe und prinzipielle Verarbeitungsvorgänge hervor.
Datenspeicher		Speicher, in dem ausschließlich Daten gespeichert sind (beim Mikrocomputer meist RAM, gelegentlich ROM).
DA-Wandler	DA-Converter	Digital-Analog-Wandler: Schaltung, die aus digitalem Code Analogsignale erzeugt.
Debugging	„Ent-Wanzen"	Fehlersuche und Beseitigung von Fehlern, insbesondere in Computer-Programmen.
Decoder	Dekodierer	Einrichtung/Schaltung/Baustein zur Umwandlung einer codierten in eine andere (von Ihnen lesbare) Information.
Digitaltechnik		dort sind für alle Signale nur zwei Pegel definiert (Low und High).
dynamischer Speicherbaustein		Speicherelement, bei dem die in Form von Ladung gespeicherte Information zyklisch „aufgefrischt" werden muß.

E

„Enable"-Signal		„Freigabe"-Signal
Encoder	Kodierer	Einrichtung/Schaltung/Baustein zum Umwandeln von Informationen.
Entwicklungssystem		Computersystem (Hard- und Software), das zur Entwicklung von Anwendersystemen dient (Programmierung, Übersetzung, Debugging, Simulation).
EPROM		(Erasable-Programmable ROM): Festspeicher, dessen gesamte Information mit UV-Licht löschbar ist. Der Speicherbaustein läßt sich dann wieder neu programmieren.

Europakarte		Leiterplatte in genormtem Format, 10 cm x 16 cm.
Execution Time	Ausführungszeit	Zeit, die zur vollständigen Ausführung eines Befehls benötigt wird.

F

Flipflop		bistabiles Speicherelement zur Speicherung von 1 bit.
Floppy Disk		relativ schnelles, externes Speichermedium mit wahlfreiem Zugriff, das sich besonders für den Einsatz in der Mikrocomputer-Technik eignet. Als Datenträger dient eine „Scheibe" aus magnetischem Werkstoff (wie Tonband).
flüchtig	volatile	Eigenschaft eines Speicherinhalts, bei Ausfall der Versorgungsspannung verlorenzugehen.
Fortran		höhere Programmiersprache für technisch-wissenschaftliche Programme (Formula Translating Language).

G

Gatter	Gate	Schaltung, die mindestens zwei digitale Signale (binäre Variablen) miteinander verknüpft.

H

Hardware		Sammelbegriff für Bauteile und Geräte.
Hardwired Logic	Festverdrahtete „Logik"	Digitalschaltung, aus Standardbausteinen aufgebaut.
HIGH		Logikpegel (entspricht bei positiver Logik der „logischen" Eins).

I

IC	IS	Integrated Circuit, Halbleiterschaltung in einem Gehäuse.
Indizierte Variable		eine Variable A mit dem Index N: A (N).
integer		heißt zu deutsch: „Ganze Zahl".
Interface	Anpassungsschaltung	elektronische Schaltung, die zwei Geräte oder Bausteine einander anpaßt.
Interrupt	Programmunterbrechung	momentan in Arbeit befindliches Programm wird unterbrochen und eine Interrupt-Service-Routine bearbeitet. Danach wird das unterbrochene Programm weiterverarbeitet.
I/O-Chip	Eingabe-Ausgabe-Baustein	integrierte Schaltung, die den Datenfluß vom und zum Mikroprozessor abwickelt.

K

Kernspeicher		Speicher aus Magnetkernen aufgebaut. Sehr teuer. Nur für sehr große Anlagen.
Klarschriftleser		Eingabegerät, das Schrift in maschinenlesbaren Code umwandelt.
kompatibel	verträglich	zwei Geräte sind kompatibel, wenn sie ohne Zusatz miteinander arbeiten können.
kundenspezifischer Schaltkreis		Exklusivanfertigung einer integrierten Schaltung für einen Kunden. Nur bei großen Stückzahlen möglich.

L

LED	Leuchtdiode	(Light Emitting Diode) farbiges Licht ausstrahlender Halbleiter (Ersatz für Lämpchen).
Lochkartenleser		periphere Eingabeeinheit eines Computers zum Einlesen von (meist 80spaltigen) Lochkarten, relativ langsam.
Lochkartenstanzer		periphere Einheit eines Computers, Ausgabe von Information in Form von Lochkarten (langsam).
Logic	Logik	Kurzbezeichnung für „logische", d. h. digitale Schaltungen.
Low-Power-Schottky-TTL		Weiterentwicklung der TTL-Serie mit höherer Arbeitsgeschwindigkeit.
LSI		(Large Scale Integration) hoher Integrationsgrad (z. B. CPU, 4-K-RAM-Bausteine).

M

Magnetbandkassette		billiges externes Speichermedium mit seriellem Zugriff.
Machine language, Object code	Maschinencode, Maschinensprache	von der Maschine direkt interpretierbare Anweisungen.
Maskenprogrammierung		Methode, den Speicherinhalt eines ROM schon beim Hersteller durch eine kundenspezifisch angefertigte Maske festzulegen.
Massenspeicher		Speicher für große Datenmengen (im Gegensatz zum Arbeitsspeicher).
Mikrocomputer	Microcomputer	Computer, dessen Zentraleinheit ein Mikroprozessor ist.
Minicomputer		Computer mit meist 16 bit Wortlänge, zur Bearbeitung weniger bis mittelmäßig komplexer Probleme geeignet.
Mnemonic Code	Mnemonischer Code	leicht zu merkende, alphanumerische Kürzel für Befehle (im Gegensatz zum Dualcode oder Zahlencode, der schwerer zu behalten ist).

MNOS		Metal Nitride Oxide Semiconductor: im Entwicklungsstadium befindliche MOS-Technologie, die Ladungsspeicherung über sehr lange Zeit ermöglicht. Interessant für nichtflüchtige RAMs.
MOS		Metal Oxide Semiconductor: Halbleitertechnologie, die sehr hohe Schaltungs-Eingangswiderstände ermöglicht.
MPU		Mikro Processor Unit: Mikroprozessor.
MSI		Medium Scale Integration: mittlerer Integrationsfaktor (z. B. 4-bit-Zähler-Baustein).

N

n-Kanal-MOS		MOS-Technologie, die mittelhohe Schaltgeschwindigkeiten zuläßt.

O

Off-Line		Form des Verkehrs mit einem Computer, bei dem der Benutzer nicht hardwaremäßig mit diesem verbunden ist, sondern der Verkehr über Datenträger abgewickelt wird.
On-Line		Form des Verkehrs mit einem Computer, bei dem das Terminal des Benutzers über eine Datenleitung direkt mit dem Computer verbunden ist (künftig auch: Telefonleitung).

P

Peripherie(geräte)		externe Geräte
periphere Speicher		Geräte zur Datenspeicherung, die an einen Computer angeschlossen werden.
PIA		Peripheral Interface Adapter: Mikrocomputer-Baustein, der den Ein-Ausgabeverkehr zwischen CPU und Peripherie abwickelt.
Pin-kompatibel		von der Anschlußbelegung (Pin-Belegung) her identische Bauteile.
P-Kanal-MOS		historisch erste MOS-Technologie, relativ langsam.
PLA		Programmable Logic Array: IS, bestehend aus zwei ROMs, z. B. zur Codewandlung.
PL/1		Programming Language 1: höhere Universal-Programmiersprache.
PL/M		Programming Language for Microcomputer: höhere Programmiersprache für Mikrocomputer, basierend auf PL/1.
Programm	Program	Folge von Befehlen.
Programm-ablaufplan	Program Flowchart	Hilfsmittel beim Computer-Systemaufbau. Aus dem Programmablaufplan geht die Struktur der Verarbeitungsvorgänge hervor.

Programmiersprachen	Programming Languages	künstliche Sprachen mit einem begrenzten Vorrat an Befehlen, die automatisch in maschinenverständliche Form übersetzbar sind.
Programmspeicher	Program storage	Speicher, in dem ausschließlich Programme gespeichert sind (beim Mikrocomputer meist ROM).
PROM		Programmable ROM: Programmierbarer Festspeicherbaustein.
Puffer(-Speicher)	Buffer	Speicher, in dem Daten vorübergehend zwischengespeichert werden.

R

RAM		Random Access Memory: Schreib-Lese-Speicher mit wahlfreiem Zugriff.
Random	Zufall	Aus einem Kollektiv von Begriffen (Zahlen, Strings) wird einer zufällig ausgewählt.
Random Access	wahlfreier Zugriff	jede Speicherstelle kann direkt angesprochen werden, also ohne daß andere Speicherstellen zuerst gelesen werden müssen.
Real-Time	Echtzeit	Arbeitsweise eines Computers: Im Echtzeitbetrieb ist er direkt in einen Prozeß einbezogen, auf den er Einfluß nimmt bzw. von dem er beeinflußt wird.
Rechenwerk	ALU	Teil einer CPU, in dem arithmetische und logische Operationen ausgeführt werden.
Refresh	auffrischen	zyklischer Adressiervorgang bei dynamischen Halbleiterspeichern zum Ausgleich von Ladungsverlusten durch Leckströme.
Register		kleine, schnelle Zwischenspeicher (meist in der CPU).
ROM		Read Only Memory: Festspeicher mit wahlfreiem Zugriff.

S

Schnittstelle	Interface	pegel- und ablaufmäßig genormter Anschluß zwischen zwei Geräten.
Schreib-Lese-Speicher	Read/Write-Memory	Speicher, dessen Informationsinhalt unter Rechnerkontrolle verändert werden kann.
Second Source	Zweitlieferant	Hersteller, der ein pin-kompatibles Bauelement aus unabhängiger Fertigung liefert.
Software		Sammelbegriff für alle Arten von Programmen.
Speicher	Storage	Medium, das Informationen über einen beliebig langen Zeitraum festhalten kann.
Speicherorganisation		Anordnung der Speicherzellen (wie viele Bits stehen pro Adresse parallel zur Verfügung).
SSI		Small Scale Integration: niedriger Integrationsgrad, z. B. Gatterbausteine.

Stack	Stapelspeicher	vom Programmierer definierter Speicherbereich außerhalb des Steuerwerks, der mit Spezialbefehlen anzusprechen ist.
Stack-Pointer	Stapelzeiger	Register, in dem eine Stack-Adresse gespeichert ist.
Statischer Speicherbaustein		Bauelement, dessen Informationen nach dem Einschreiben auch ohne zusätzliche Maßnahmen erhalten bleiben (außer bei Stromausfall).
SN ERROR	Syntax Error	Fehlermeldung des Computers: Verstoß gegen die Regeln bei der Eingabe.
Speicher		Medium, in dem Information über längere Zeit verfügbar gehalten wird.
Steuerwerk	Control Unit	Teil einer Computerzentraleinheit, der die Ausführung sämtlicher Befehle kontrolliert.
String-Variable		eine Variable, deren „Wert" eine Zeichenkette ist.
Subroutine	Unterprogramm	Programmabschnitt, der im Verlauf eines Hauptprogramms mehrfach durchlaufen wird, aber nur einmal gespeichert ist.
System-Analyse	System Analysis	Phase in der Projektplanung und Bearbeitung, in der die Struktur des Problems analysiert wird.

T

Takt	Clock	praktisch alle existierenden Computer sind „Synchron-Maschinen", d. h. ihre Operationen erfolgen in einem festgelegten, von einem Taktgenerator bestimmten Zeitraster.
Taktgenerator		Schaltung/Baustein zur Erzeugung des computerinternen (Arbeits-)Taktes.
Terminal	Datenendgerät	Gerät zur Dateinein- und/oder Datenausgabe.
Time Sharing	Zeitscheibenverfahren	Verfahren, bei dem mehrere Benutzer On-Line auf eine Großanlage zugreifen können, wobei jeder Benutzer den Eindruck hat, daß ihm allein die Anlage zur Verfügung steht.
Tri-State-Gatter		Digitalschaltung, geeignet zum Anschluß an Busse. Nur aktivierte Elemente bestimmen die Bus-Information. Schneller als Open-Collector-Gatter (Gegentakt).
TTL		(Transistor Transistor Logic) mittelschnelle digitale Standardbausteinserie mit 5 V Versorgungsspannung.
TTY		Fernschreiber (Teletype), meist mit angeschlossenem Lochstreifenleser/-stanzer.

U

Unterprogramm	Subroutine	Programmabschnitt, der im Verlauf eines Hauptprogramms mehrfach durchlaufen wird, aber nur einmal gespeichert ist.

V

Variable		eine Variable kann in einem Computerprogramm einen vom Programmierer definierten Wert erhalten.
VLSI		(Very Large Scale Integration) in den 80er Jahren zu erwartender, sehr hoher Integrationsfaktor (z. B. vollständiger Microcomputer auf 1 Chip).
volatile	flüchtig	Eigenschaft eines Speicherinhalts, bei Ausfall der Versorgungsspannung verlorenzugehen.

W

wahlfreier Zugriff	Random Access	jede Speicheradresse kann direkt, also ohne daß andere Speicherstellen zuerst gelesen werden müssen, angesprochen werden.
Wortlänge		Anzahl der Bits, die zusammenhängend verarbeitet werden können. Bei Home-Computern 8 Bit.

Z

Zentraleinheit	CPU	Central Processing Unit: Rechenwerk + Steuerwerk eines Computers.
Zufallsgenerator		Einrichtung/Programmvariante, die (meist Zahlen) zufällig ausgibt.
Zugriff		Möglichkeit, eine bestimmte Speicherstelle anzusprechen (zu adressieren) und zu lesen.
Zuweisung		beim Programmieren mit Variablen wird diesen ein bestimmter „Wert" zugewiesen.

Weitere Franzis-Computer-Fachbücher

Basic für Mikrocomputer

Geräte – Begriffe – Befehle – Programme. Von Herwig **Feichtinger.** – 2., neu bearbeitete Auflage. 264 Seiten, 42 Abbildungen. Lwstr-kart. DM 28.–

ISBN 3-7723-6822-0

Dieses praxisorientierte Buch ist Einführung und Nachschlagewerk zugleich. Begriffe aus der Computer-Fachsprache wie ASCII, RS-232-Schnittstelle oder IEC-Bus werden ebenso ausführlich erläutert wie alle derzeit üblichen Befehlsworte. Marktübliche Basic-Rechner werden einander gegenübergestellt, um vor dem Kauf die Wahl zu erleichtern und um das Anpassen von Programmen an den eigenen Rechner zu ermöglichen. Schließlich findet der Leser handfeste Tips für das Erstellen eigener Programme und Beispiele fertiger Problemlösungen für typische Anwendungsfälle.

Erfolgreicher mit CBM arbeiten

Für alle CBM-Anwender eine verständliche Einführung in die Maschinensprache. Von Dipl.-Ing. Franz **Wunderlich.** – 2., völlig neu bearbeitete und erweiterte Auflage. 248 Seiten, 11 Abbildungen. Lwstr-geb. DM 42.–

ISBN 3-7723-7052-7

CBM-Anwender mit Basic-Erfahrung holen mit diesem Buch mehr aus ihrem Computer heraus. Sie können nämlich mit dem Programmieren in der Maschinensprache beginnen. Zunächst wird ihnen beigebracht, wie ein 6502 programmiert wird. Zahlensysteme, Speicherkonzepte, Adressierung und Befehlssätze werden behandelt und gewinnen Klarheit. Im Hauptteil wird speziell die geräteabhängige Software der Commodore-Serie CBM abgehandelt. So wird beschrieben, wie Interpreter, Betriebssysteme, Monitor und Peripherie arbeiten. Natürlich bringt der Autor auch fertige Programme und viele CBM-spezielle Anwenderbeispiele im Anhang.

Basic-Interpreter

Funktionsweise und Implementierung in 8080-/Z-80-Computern. Von Rolf-Dieter **Klein.** – 2., verbesserte Auflage. 178 Seiten, 43 Abbildungen. Lwstr-geb. DM 36.–

ISBN 3-7723-6942-1

Wie man 8080- oder Z-80-Systeme nachträglich mit einem Basic-Interpreter ausrüsten kann, beschreibt dieses Buch. Dabei werden mehrere Ausführungen erörtert und beschrieben. Die beiden interessantesten sind: Ein Tiny-Basic-Interpreter und ein komfortabler 12-KByte-Basic-Interpreter. Die Krönung bildet die ausführliche Beschreibung eines Basic-Interpreters für den 16-Bit-Prozessor Z8000.

Basic für Aufsteiger

Der sichere Weg zum fortgeschrittenen Basic-Programm. Von Rudolf **Busch.** – 229 Seiten mit 44 Abbildungen und 12 Tabellen. Lwstr-geb. DM 39.–

ISBN 3-7723-7281-3

Basic für Aufsteiger – das ist das Basic, das nicht mehr in den Handbüchern steht. Wer dieses Buch intus hat, der besitzt einen anspruchsvollen Basic-Wortschatz, der sich sehen lassen kann.

Die Schwerpunkte liegen in der Aufbereitung aller mathematischen Funktionen, die Basic bietet. Es folgen jene, mit denen Strings unter allen nur denkbaren Gesichtspunkten bearbeitet werden können. Das Buch schließt ab mit einer Fülle von Programmen aus Mathematik, Geometrie, Betriebswirtschaft und Datenverarbeitung.

Der Autor greift seine Beispiele aus dem täglichen Leben, kleidet sie oft in lustige Geschichten, denn so lernt der Leser am besten, was er tut und warum er es tut.

Mikrocomputersysteme

Selbstbau – Programmierung – Anwendung. Von Rolf-Dieter **Klein.** – 4., verbesserte Auflage, 159 Seiten, 134 Abbildungen und 12 Tabellen. Lwstr-geb. DM 36.–

ISBN 3-7723-6384-9

Kaum zu glauben, daß ein Mikrocomputer im Selbstbau hergestellt werden kann! Daß dieses Vorhaben glückte, hat der Autor bewiesen. Wie ein hinreichend ausgebildeter Elektroniker das nachvollziehen kann, wird in dem Buch hier dargestellt.

Mikrocomputer selbst gebaut und programmiert

Vom Bauelement zum fertigen Z-80-Computer. Von Rolf-Dieter **Klein.** – 2., neu bearbeitete und erweiterte Auflage. 419 Seiten mit 370 Abbildungen. Lwstr-geb. DM 38.–

ISBN 3-7723-7162-0

Vier gewichtige Punkte unterscheiden dieses Buch von den herkömmlichen Titeln über die Mikrocomputerei:

1. Die Hardware wird durch ein Experimentalsystem, bestehend aus kleinen, selbst zu bauenden Modulen im wahrsten Sinne des Wortes begriffen.
2. Alle Schaltungen sind mit einer ausführlichen Aufbau- und Testanleitung, einschließlich der Platinenvorlage, ausgestattet.
3. Die Software-Unterrichtung erfolgt mit genau auf den Z-80 abgestimmten Programmierschriften. Kein Befehl zuviel, keiner zu wenig.
4. Die konsequente Systematik und die pädagogisch sinnvollen Kontrollfragen machen den Band zu einem vorzüglichen Lehr- und Arbeitsbuch.

Wer das Buch durchgearbeitet hat, der hat die Computer-Grundlagen in sich aufgenommen und parallel dazu einen leistungs- und ausbaufähigen Mikrocomputer selbst gebaut.

Liefermöglichkeiten und Preisänderungen vorbehalten!

Franzis-Verlag München

Busch
Basic für Einsteiger

Nie wird der zweite Schritt vor dem ersten gemacht. Das ist der unbändige Vorzug dieser Programmierfibel. Ein weiterer kommt hinzu und potenziert ihre Brauchbarkeit.

Die meisten Kapitel sind „zweiseitig" aufgebaut. Das heißt, auf der linken Seite steht, was dem Computer eingegeben wird und wie er darauf reagiert. Auf der rechten Seite werden die verwendeten Befehle mit Kommentaren erläutert.

Auf diese simple Art und Weise wird die Programmiersprache Basic gelernt und ihre Sprachelemente erläutert. Der junge Programmierer wird dahin geführt, selbständig seine Problemstellungen zu analysieren. Zum lauffähigen Basic-Programm ist es dann nur noch ein Sprung.

Wie heißt doch der Titel dieses Buches? Basic für Einsteiger.

ISBN 3-7723-7082-9

Franzis'